SINCLAIR QL WORLD

Volume 2, Issue 12 £2.50

Machine Code for Beginners TRAP CALLS

Reach for the SKY!

Airliner flightdeck for the QL

THE ULTIMATE SQEAK

The DIY Mouse adapted to other programs.

ISSN 0951-9335 12

Fed up of DIGITAL PRECISION telling you how very good their software is?

✓ "As you might surmise by this time, I am impressed by QMATHS's abilities. Having noted that DIGITAL PRECISION's advertising tends to be loaded with superlatives (incredible, ultimate, superb come to mind), I had approached this evaluation with some scepticism. **That scepticism has vanished.**" > INTERNATIONAL QL REPORT (IQLR, available from Miracle Systems) May/June 1993 issue, Official Review by M.Laverne commissioned by IQLR (who bought their review copy of the program).

✓ "PERFECTION is an exciting, full-flavoured, general purpose word processor of incredible capacity... PERFECTION has now been outshone by the recently released PERFECTION SPECIAL EDITION... The discoveries began to trip over themselves as PERFECTION SE responded to the keyboard with unexpected speed and intelligence... PERFECTION SE is blindingly fast at most things, and you are never left waiting for it. PERFECTION is everything that Quill never became: easy to use, very flexible, loaded with genuinely useful features, cleanly multi-tasking, capacious and incredibly fast. The SPECIAL EDITION offers 12 cylinder power and luxury to an already impressive package." > SINCLAIR QL WORLD magazine Official Review, April 1993 issue, by **THE** Mike Lloyd of Keyword Index / New QL User Guide fame.

✓ "I find PROFESSIONAL PUBLISHER an outstandingly good program that really does allow highly professional documents to be produced. For your interest I have included a few samples of work done for school using a combination of PERFECTION, PROFESSIONAL PUBLISHER, QUICKLASER and EYE-Q. You will be pleased to know that the quality has been rated so highly that people do not believe it can really have been done with just a QL... I must stress that I am already highly impressed with, and very satisfied by, the performance of PROFESSIONAL PUBLISHER and all the other DP programs that I use... I seem to learn something new that can be done almost each time I use the program. Very many thanks for helping to keep the QL ahead of the field." > Martin J Neave, Headteacher, Watton County Jnr School, Brandon Rd, Watton, Norfolk, IP25 6AL (unsolicited letter dated 18 May 1993 ordering more programs: Mr Neave had paid full price for everything).

✓ "LIGHTNING SPECIAL EDITION accelerates QL operation as nothing else does... more than 10x is achievable and 2x-4x is typical... I could not fault LIGHTNING SPECIAL EDITION on anything. It is a clear winner and a best buy at £49.95." > SINCLAIR QL WORLD magazine Official Review, April 1990 issue, by Ron Massey, who wrote EDITOR (bought full price) was "Superb" in an earlier review.

✓ "PERFECTION is well named" > R.H.Petford, Kingston Hill, Surrey, KT2 7LJ (unsolicited letter received May 25, 1993: another full price purchaser & upgrader).

✓ "When my ideal program finally arrived in the form of PROFESSIONAL PUBLISHER, it surpassed all my expectations... PROFESSIONAL PUBLISHER (is) in a class of its own, and makes it the only QL desktop publishing program for the very serious user... Until Digital Precision released PROFESSIONAL PUBLISHER, my opinion was that the use I could make of desktop publishing was mainly restricted to short documents... PROFESSIONAL PUBLISHER is a very versatile program... The illustrations for this series of articles have all been produced on PROFESSIONAL PUBLISHER... My printer is a BROTHER 9-pin dot matrix printer. It does illustrate the very high quality that can be obtained from PROFESSIONAL PUBLISHER even when using a simple printer." > SINCLAIR QL WORLD magazine Guide to desktop publishing ("A Question of Dots"), January 1992 to December 1992 issues: the reviewer had bought PROFESSIONAL PUBLISHER, PERFECTION SE, FONT ENLARGER, TOOLBOXES, QUICKLASER etc from Digital Precision all at full price.

✓ "I am aware that over the years Digital Precision has given considerable support to the QL scene but seldom, if ever, can there have been such estimable service as I recently encountered with PERFECTION PLUS." > The Hon. W.D.R. Spens, Bridgewater, Somerset, TA5 1HG, QUANTA magazine, March 1992 issue. Mr Spens has bought a lot of his software from Digital Precision, all at full price of course.

✓ "The Digital Precision Desktop Publisher was rightly hailed as an extraordinary programming achievement when it was released two years ago. Mike Lloyd casts a professional eye over Digital Precision's latest page-making blockbuster *(PROFESSIONAL PUBLISHER)* and finds plenty to be pleased about... there is unlikely to be a single program of such magnitude and quality *(as PROFESSIONAL PUBLISHER)* written for the Sinclair QL." > SINCLAIR QL WORLD Official Review, August 1989 issue, by M.Lloyd, who personally bought all this at full price.

✓ "EDITOR is a liberation. After Quill, it was like jumping from an aquarium into the sea. It has become part of my professional life... Everyone is now writing about the excellence of PERFECTION. I have not tried it, not having any perceived need for it *(having EDITOR)*" > Suzanne Cronje, QUANTA magazine, May 1992 issue, page 2. Ms Cronje naturally had paid the full price for her copy of EDITOR SE etc.

✓ "I have found (PERFECTION) to be simply excellent, fast, packed with features and very well thought out. I can find little to say that will convey just how good this program is, except to quote Digital Precision's own advertising: PERFECTION will blow your socks off. PERFECTION is the program that Quill users have been waiting for." > SINCLAIR QL WORLD's first Official Review, May 1991 issue.

✓ "Digital Precision (DP) decided to begin work on a replacement for Quill which would be very quick, simple to use and contain lots of excellent features - something upon which DP have built a very strong reputation in the QL market... Overall, the speed-up *(of just the first release of PERFECTION - it is much faster now)* on even a humble QL with Trump Card is amazing when compared with Quill (or any other word processor). On top of this, the program provides many excellent and well thought out features, each of which is easy to use... *(it)* is certainly years ahead of the competition on the QL (and even on many PCs)." > R.Mellor, c/o CGH Services, Cwm Gwen Hall, Pencader, Dyfed, SA39 9HA; Official Review of the very first version of PERFECTION in QL TECHNICAL REVIEW issue 7: and the reviewer personally bought his own copy of this program, and many others at full price, from Digital Precision. Earlier QLTR reviews pronounced LIGHTNING (just the standard version) superior to the competition and ADVENTURE CREATION TOOL excellent.

✓ "PERFECTION SE is superb!! With Gold Card, it puts life in the fast lane. Thanks." > Leonard Singleton, Bletchley, MK3 6BP, June 1993, a full price purchaser (=fpp).

✓ "As a recent user of PERFECTION PLUS SE, may I add my thanks and praises to the ones I am sure you have already received... keep up the excellent work." > R Slawson, East Molesey, Surrey, KT8 0BP(unsolicited letter from full price purchaser).

✓ "At about 360,000 words, the Mega SPELLCHECKER dictionary does not have much competition, on any computer! (Spellchecking) is about four times as fast as the best figures I have seen with other checkers on QL and PC." > SINCLAIR QL WORLD magazine official review of PERFECTION spellchecker, September 1992 issue, by Bryan Davies of Troubleshooter repute (review copies of all the competing products supplied to SINCLAIR QL WORLD by their respective publishers).

✓ "In the past I have purchased a number of your programs and have never failed to be impressed by the quality of both product and documentation. (So) please send a list of your current products." > V.Negri, Hempton, Norfolk, NR21 7LF, June 1993, fpp.

✓ "This is my first letter with PERFECTION SE. I must say I'm impressed with it and it is certainly fast. Hooray, goodbye to Archive!" > P.H.Heilbron, Reigate, RH2 0DJ, a full price purchaser now using PERFECTION to replace not only Quill but Archive too.

✓ "I have been using PROFESSIONAL PUBLISHER for about eighteen months now... what you can do with it is colossal... I got Digital Precision's QUICKLASER. The results are as good as *(Digital Precision)* says in its advertisements..." > P.Hamill, Peterborough, Cambs, PE8 6RH, QUANTA magazine, Volume 9 issues 4/12. Mr Hamill (full price purchaser) then makes suggestions to users re optimal page sizes.

✓ "Once again I would like to say thank you for your help. I would like to tell the world what nice guys you are but unfortunately I have no contact with the outside world." > J.Bailey, Godshill, Ventnor, PO38 3JJ (full price purchaser, 24 May 1993).

✓ "PC CONQUEROR GOLD SPECIAL EDITION is an excellent product, accompanied, as so often with Digital Precision software, by a comprehensive and informative manual. The program does a difficult job, and does it well... Overall, this program is much faster, more compatible and capable..." > SINCLAIR QL WORLD Official Review, March 1993 issue, by M.Knight (bought many DP programs full price).

✓ "With printing of the quality that this page bears witness to, I am a very satisfied PERFECTION user. I hope that you continue to provide the software innovation and the accessible backup which is great. So, thank you very much again and may I wish you every good fortune." > P.Stewart, Temple, London, EC4Y 9BE, 10 May 1993, fpp.

✓ "Many thanks for the update of PERFECTION SPECIAL EDITION. I am suitably impressed. Congratulations on producing the only word processor that I know that offers the best of all worlds as far as formatting is concerned. After Quill, PERFECTION is like a breath of fresh air." > Geoff Wicks, 1097HL Amsterdam, Netherlands (unsolicited letter dated 13 June 1993: all software including LIGHTNING PERFECTION SE, PRO PUBLISHER, CONQUEROR SE etc. purchased at full price).

✓ "All I can say about QMATHS is: **WOW!**" > Robin Wyke-Holloway, Salisbury, SP5 4WG (unsolicited letter received April 1993: Mr Holloway is a full price purchaser).

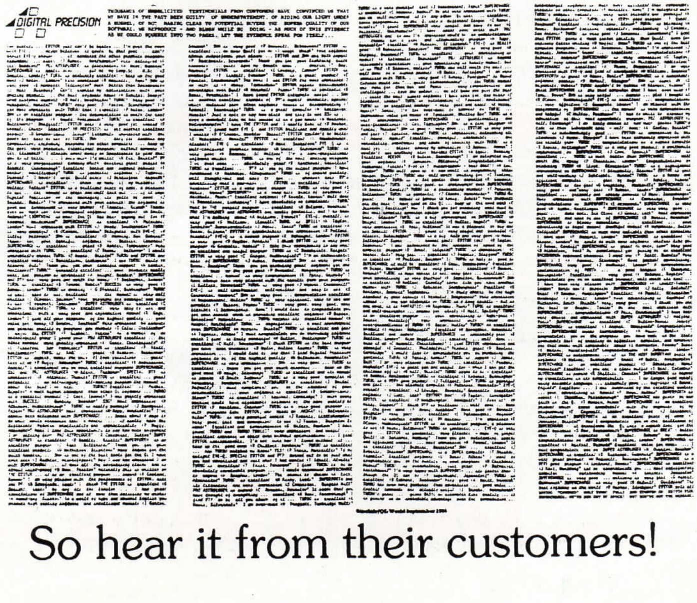
✓ "I have had PERFECTION from the early days and have had many hours pleasure finding out more and more of its brilliant features. May I offer my congratulations on such an easy to use program which does everything I want - and more besides." > F.Merrison, Pinner, HA5 5AZ, fpp, thanking us for fixing a printer problem he'd had.

✓ "Having used a range of desktop publishers on the Atari ST & Amiga, I admit I am very impressed with the superior performance of PROFESSIONAL PUBLISHER. It contains everything required" > SINCLAIR QL WORLD January 1989 issue, article entitled "Six of the Best" which also praised five other new Digital Precision programs.

✓ "I find it difficult to express my gratitude for the speed of your response, and for the opportunity to see inside two excellent programs which I have long enjoyed using." > Richard Walker, Enniskillen, BT74 7LG, full price purchaser and QL expert.

✓ "May I take this opportunity to say that I have, in the past, found the software you have supplied me with (LIGHTNING etc.) to be of extremely high standard, on a par with that found in industry-standard PC packages. Keep up the good work. Without your quality software, I would be forced to abandon the QL and go to a PC." > G. Reynolds, Crosby, Liverpool, L23 0SS (unsolicited letter dated April 2 1993, placing a further order for DP software: all programs old & new were purchased at full price).

✓✓ *This is but a casual selection, drawing only on extracts from recent letters and reviews. We could locate >1000 complimentary communications but we'd prefer to spend our time producing new programs! We refer potential customers (existing users of DP products already know how good they are) to pages 18/19 of the September 1988 issue of Sinclair QL World, which contained well over three hundred other unsolicited quotations from happy Digital Precision customers. That collection covered only three programs (and that too only partly - we ran out of space) and predated the release of what many consider to be our best software (LIGHTNING SE, PERFECTION SE, PROFESSIONAL PUBLISHER, CONQUEROR SE etc). We reproduce those pages below, duly reduced to fit. If you want a readable copy, consult the relevant back issue or send us an SAE or ask for a full-sized copy while ordering from us...*


DIGITAL PRECISION
THOUSANDS OF UNSOLICITED TESTIMONIALS FROM CUSTOMERS HAVE CONVINCED US THAT WE HAVE IN THE PAST BEEN GUILTY OF UNDERSTATEMENT, OF HIDING OUR LIGHT UNDER A BUSHEL, OF NOT MAKING CLEAR TO POTENTIAL BUYERS THE SUPERB QUALITY OF OUR SOFTWARE. WE REPRODUCE - AND BLUSH WHILE DOING - AS MUCH OF THIS EVIDENCE AS WE COULD SQUEEZE INTO TWO PAGES. LET THE EVIDENCE SPEAK FOR ITSELF...

So hear it from their customers!

PERFECTION SPECIAL EDITION

AN EXCITING NEW DEVELOPMENT - Version 5!

In the case of many word-processing objectives, the best way to implement them is pretty clear. There are some areas, however, where individual tastes and preferences can differ very widely. One such area is the reformatting of text - the adjustment of previously entered text to conform to margin, indentation, justification and pagination settings after you go back (or forward!) to it and make alterations, either by hand (by typing and/or deleting) or by using individual or global search and replace. When new text is being entered at the foot of the document or at the end of the current paragraph, all word-processors behave virtually identically, obeying the current settings - it is in the matter of amending existing text (inserting, changing or deleting) where conflicting philosophies apply. Text-handlers differ in their treatment of this: Editor, Wordperfect, text^{87}, Quill, AmiPro & Word all behave differently.

Editor, Spy, most versions of Wordstar, and all technical editors leave all reformatting to you. While at first this may seem harsh, this manual mode gives you a lot of control, makes the handling of tables and other technical applications better (do you really want to reformat that BASIC program into a single paragraph?!?), and is easy on the eye. But you must remember to reformat as the program won't, and this may be an annoyance. If you move away and forget to clean up, your printout will probably be incorrect.

Wordperfect **will** auto-reformat, but generally only when you move the cursor from the line containing the change. Changes you make while your cursor is within the line will only cause the line to contract or expand up to the margin. This too is easy on the eye, but there is the drawback that the overall picture of the page may be inaccurate while you are inserting or amending text, and that when you move the cursor away (and hence trigger the auto-reformat), you may not notice any undesirable effects caused (e.g. widows, orphans, inappropriately positioned page or line breaks).

QL Quill auto-reformats, but because of its slowness it uses a trick: often when you start inserting within the middle of a paragraph, Quill splits the para in two and creates temporary blank lines to separate the parts. This means Quill does not need to reformat until you have finished amending. What you type appears at the end of the first part of the paragraph. This has the advantage and disadvantages of the Wordperfect method, but additionally the split can be a bit disconcerting and the screen display can be grossly wrong during the editing. Also, we know of a bug that causes a line to be shown twice on the Quill screen while it is only really present once: you will regret it if you delete the apparent duplicate as an unduplicated line will get deleted without warning.

Word (a fine PC Windows program) auto-reformats in situ, in real time, as-you-type. But if you have a long complex para and you are editing near the top of it, you may notice the time taken for the reformat *even on a 486/66MHz* (QL users should note that this is >20 times faster than a Gold Card i.e. about the speed we expect from a *fully tweaked* QXL). Also, cursor movement will appear to some as a bit erratic (which is hard on the eye) especially if right justification is on or if the on-screen fonts are proportional. It can also be quite distracting to keep seeing the ripple effect of changes as text on lower lines is reformatted. AmiPro is somewhat better in this respect as there is a small delay (almost a second) before AmiPro refreshes lower lines on the screen: easier on the eye.

The new release of **PERFECTION SPECIAL EDITION**, version 5, gives the user <u>the best of all worlds</u>, by combining the best of all the above methods and avoiding all the drawbacks. The user is given the opportunity both to pre-configure and to adjust at will from inside the program, the desired auto-reformatting behaviour. The options are to either select Never (giving Editor-like action for technical users: this is what all previous versions did, where you had to press a key to get the para to reformat after re-editing it), Instant (giving in-situ real-time automatic reformatting as-you-type, as does Word) or User-delay, the most flexible setting of all (giving slightly delayed updating of lower lines of text, like AmiPro, but also - and unlike AmiPro - giving you, the user, full control over how long the delay is). No other w.p. is this able.

On User-delay the user is free to set any delay from 0.1 seconds to 99.9 seconds in 0.1 second steps. About 1-2 seconds is best for slow typists, and 1.5 seconds is thus the default. This means that you are not hassled by continuing screen changes on lines below the one you are editing and concentrating upon, or shufflings around on the current line caused by right justification etc. So the Word disadvantage (much more noticeable on slower hardware) is avoided, without recourse to the Quill temporary blank line nuisance. When you pause in your typing for longer than the set delay, **PERFECTION SPECIAL EDITION** (**SE**) automatically tidies up, without you having to do anything (getting around the Wordperfect and Quill drawback of making you mentally adjust for the screen remaining occasionally out-of-sync with reality).

If you are a reasonably fast typist, you can experiment with shorter delays (say 0.5 seconds). If you are a speed demon, set the delay to 0.1 seconds and see if you can ever manage to "get ahead" of the program! Settings of under 0.3 seconds are indistinguishable from 'Instant', when reformatting always keeps pace.

On the User-delay setting **PERFECTION SE** will, as does Quill and Wordperfect, auto-reformat *instantly* (no matter how long a delay you have set) **if** you either navigate off the line **or** invoke **any** menu or direct command (including Save, Export etc.). This means that you are never left with the document "wrong".

There are many other improvements in this release of **PERFECTION SE**. One in a similar area is with SHIFT/CAPS, the one (out of five) manual reformatting commands that allowed reformatting of a para from the current line onwards without affecting previous lines. SHIFT/CAPS will now additionally obey the indent margin (which matters if the cursor is on the first line of the para) and, more significantly, it will leave the cursor position unaltered within the text (previously, it used to move the cursor to the start of the next para). Other reformatting commands are unaltered, so you can still step through paras reformatting easily.

PERFECTION SE v5 costs £99.95, or £139.95 in **PLUS SE** incarnation (i.e. with spellchecker, dictionaries & maintenance programs), less discounts that can total 40%. There is no special upgrade price to v5 for existing **SE** owners - only DP's usual reasonable £10 update charge (but as an offer to ***IQLR*** readers, open for four weeks from the date of publication of this issue, existing **SE** or **PLUS SE** owners can get the upgrade totally free provided they order other DP programs of total value (after all discounts) exceeding £25). To upgrade from the STANDARD version of **PERFECTION** costs, as with all upgrades, the difference in price plus just £10, i.e. £50. The user should not return any documentation, just the one master disk. Remember special deal prices, which give discounts of up to **25%** if more than one program is purchased (or upgraded) at the same time (do you have LIGHTNING SE?). To get the very best out of **PERFECTION SE**, use it with PROFESSIONAL PUBLISHER (and perhaps with attendant TOOLBOXes and FONT ENLARGER), when you can output text to any number of shapes of any desired complexity (not just boring columns!) throughout maintaining pixel proportional spacing and having thousands of fully WYSIWYG fonts to choose from, whatever your printer....

All trademarks are acknowledged as belonging to their respective owners.

OTHER SPECIAL PROGRAMS FROM DP

PC CONQUEROR GOLD SPECIAL EDITION The rave review on pages 16 to 19 of March 1993 QL World really says it all: "an excellent product", "much faster, more compatible and capable than its predecessor". There are many extra features too. You can also get DR-DOS v6.0 (with Netware Lite free), which is the best DOS of all. And if you are buying or have bought this DOS from us, you can buy preconfigured DOS pseudo hard disks (on ED diskette) for £15 each (specify if you want compressed i.e. 6Mb capacity, or 3Mb: or have one of each for £25).

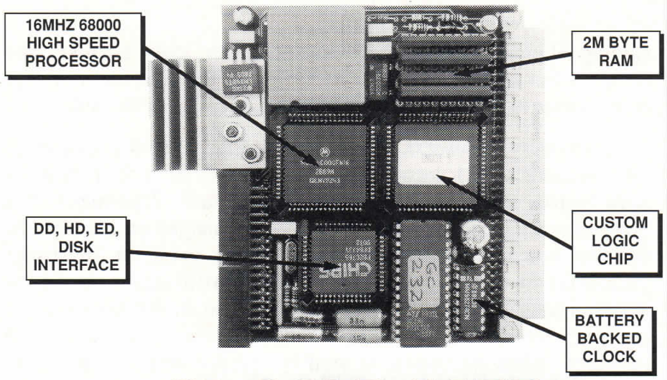
QMATHS MATHEMATICAL SYSTEM PART TWO A superb companion to QMATHS, with maths, stats, Abacus stuff, expression evaluation, terrain plotting, the fastest Mandelbrot routines and much more. Note the special price for 1+2.

TRANSFER UTILITY SPECIAL EDITION Copies and transfers, with optional sorting, case-changing, formatting, statistics and more.

QUICKLASER Superb print output from PRO PUBLISHER to HP Deskjets, Laserjets (the latter with 1Mb of RAM or more) and all compatibles. QUICKLASER costs just £19.95 all inclusive.

LIGHTNING SPECIAL EDITION GOLD CARD VERSION Optimal speed from higher specified QLs - GOLD CARD, QXL, ST/QL, Thor XVI etc. Free upgrade from standard version if you return ROM + disk and are ordering something else at the same time, else £10 charge.

MIRACLE SYSTEMS



QL GOLD CARD

£225 inc. (£200 outside EC)

This is the expansion that has been revolutionising the QL. It is very easy to fit - it simply plugs into the expansion port at the left hand of the QL - and once fitted it will instantly increase the execution speed of the QL by about 4 times due to the presence of a 16MHz 68000 on board. There is 2M of fast 16 bit RAM of which QDOS sees a contiguous 1920K. The remainder is used for shadowing the QL's ROM and display memory and for the GOLD CARD's own code.

There is a disk interface which can access 3 mechanisms (4 with the DISK ADAPTER) of 3 different densities, DD (double density, 720K), HD (high density, 1.44M) and ED (extra high density, 3.2M) in any mix. The disk interface connector is the same type that was fitted to the TRUMP CARD so most QL compatible disk drives can be used. Please note that DD drives still give a capacity of 720K per diskette. Our DUAL ED DISK DRIVE allows the GOLD CARD to access DD, HD and ED diskettes.

Another feature is the battery backed clock. When the QL is switched on the contents of the clock are copied into the QL's clock so that the time and date are correct. The firmware in the ROM gives the GOLD CARD all the functionality of the TRUMP CARD like TOOLKIT II and there is a sub-directory system for floppy and RAM disks.

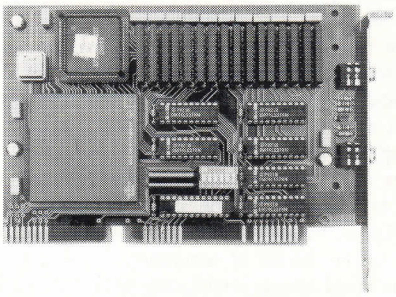
Physically the GOLD CARD is about half the size of the TRUMP CARD and so fits almost all within the QL. Its current consumption is well under the allowable maximum so no special power supply is required. The GOLD CARD comes with a 14 day money back guarantee and a 2 year warranty.

See us here:

Sunday 23rd January 1993
Georgian House Hotel
Blackrod
nr. Manchester

Saturday 12th February 1993
Scottish Workshop
Edingburgh
For venue details contact Alan Pemberton,
68 Lingerwood Road, Newtongrange,
Midlothian EH22 4QQ

THE QXL



The QXL turns the common PC into a QL compatible. The package comprises a half card that plugs into an 8 or 16 bit standard ISA slot and a diskette loaded with a QDOS compatible operating system and a Superbasic compatible interpreter. After installation simply type QXL and the PC will appear to be a QL allowing QL programs to be run from QL format diskettes.

The card itself has a 32 bit 68EC040 processor running at 20MHz which gives a good turn of speed. This processor has access to its own RAM and so performance is virtually independent of the host PC whether it has an 8088 or a Pentium. In fact the PC is used purely as an I/O system giving QL programs access to the PC's floppy disc, hard disc, keyboard, display, serial and parallel ports. The card itself has QL style network ports to allow connection to a QL network. The minimum PC specification required is an XT with EGA display and a spare standard slot.

Varying RAM sizes from 1M up to 8M can be supplied. The smaller capacities can be upgraded to the larger ones and the cost is simply the price difference. Not all the RAM is available to the user programs; the 1M equates roughly with a TRUMP CARD QL memory size and the 2M with a GOLD CARD QL.

During the lifetime of the QXL we intend to enhance the software to make use of the new hardware facilities of the PC such as SVGA graphics. As has been our policy with the TRUMP CARD and GOLD CARD we intend to provide software upgrades free of charge.

QXL prices

1M	**£295**	**(£255)**
2M	**£325**	**(£280)**
5M	**£410**	**(£355)**
8M	**£495**	**(£430)**

(prices in brackets for outside EC)

INTERNATIONAL QL REPORT (IQLR) is a regular magazine that all QL users should read. It has articles for the beginner, the advanced user and every one else in between. Also, the international flavour combined with low advertising rates makes it probably the best place to locate QL related items. IQLR is run by QL enthusiasts whose proud boast is that they have never been late with an issue. If you do not already get it then 'phone us now. One year's subscription for 6 issues to any European address is £22.00 and it's worth every penny. Subscribers elsewhere should contact SeaCoast Services, 15 Kilburn Court, Newport, RI 02840, U.S.A. direct.

VISA MASTERCARD

Tel: (0904) 423986

To place an order by phone please have your credit card ready. For overseas customers we charge the prices shown in brackets.

To order by post, please fill in the form opposite or write to us quoting your credit card number and expiry date, or enclosing a cheque payable to MIRACLE SYSTEMS Ltd.

**To: MIRACLE SYSTEMS Ltd, 25 Broughton Way,
Osbaldwick, York, YO1 3BG, U.K. Tel: (0904) 423986**

Please send me ____________________

I enclose a cheque to the value of: £ ____________________

Or debit credit card □□□□□□□□□□□□□□□□

Expires □□ / □□

Name ____________________ Signature ____________________

Address ____________________

____

QL Contents

VOL 2 ISSUE 12

Coming Soon

QL Calendar - HDD disk and new QL boxing system - WorldMap.

Editor
Helen Armstrong

Publisher
Mark Kasprowicz

Advertising Manager
Jim Peskett

Creative Director
John Stanley

Graphic Artist
Stevie Billington

Magazine Services
Linda Miller, Frances Maxwell, Pauline Wakeling

Sinclair QL World,
Published by Arcwind Ltd.
The Blue Barn,
Tew Lane, Wootton,
Woodstock,
Oxon. OX7 1HA
Tel: 0993 811181
Fax: 0993 811481
ISSN 026806X

If you have any comments or difficulties please write to the editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.
Back issues are available from the publisher price £2.50 UK, £2.99 Europe. Overseas rates on request.

Subscriptions from: Arcwind
The Blue Barn, Tew Lane,
Wootton, Woodstock, Oxon.
OX7 1HA
UK: £23.40
Europe: £32.90
Rest of World: £40.90

Reprographic Services:
Eclipse, Brook Street,
Watlington, Oxon. OX9 5JH.
Distributed by: Seymour Press Ltd., Windsor House, 1270 London Road, Norbury, London, SW16 4DH

While all reasonable care is taken in compiling QL World, the publisher and its agents assume no responsibility in affects arising therefrom. Opinions expressed are those of the authors.

QL Scene

TF Services takes the I2C Bus

Tony Firshman of **TF Services** has released the first two in a new family of I2C bus **hardware interfaces** - here is the information we promised you.

The I2C bus was designed by the Dutch electronics giant Philips to simplify hardware interfacing to computers. The Minerva MKII (RTC) rom adds a battery-backed ram and a clock using the I2C bus. Computer users can also get a cable and a 9-pin D connector to allow other I2C devices to be added. There are a wide range of interesting items from phone diallers, LCD and relay drivers, teletext chips to microprocessors and many radio and TV controllers. The Minerva will not, however, allow the use of microprocessors and other devices needing dual bus masters with the QL.

TF Services' interfaces are based on the parallel and analogue I2C chips. The interfaces themselves are in small boxes with high quality double-sided circuit boards. They are linked to each other by 15-pin cable-less D connectors, and each interface contains two I2C chips, each of which can have one of 8 addresses (set by DIP switches). Power and signal lines are taken from the Minerva connector, and at least four interfaces can be powered without using an external power supply.

The software which comes with the connectors is designed to make reading and writing to the devices easy. Only the device address (0 to 7) is needed to read from either device, and to write, only the address and data are needed. The code can multitask in compiled SuperBasic programs. Machine code traps are provided for machine code programmers. Both devices are accessed at close to the maximum speed to 100 Kbits per second.

Data is only received and sent on a single data line after successful completion of a handshake protocol, which includes a unique I2C chip address.

The **ANALOGUE INTERFACE** can handle 8 bytes ADC (analogue to digital) and two bytes DAC (output). It can be used directly for such tasks as frequency analysing - one customer who works for tv transmission links is planning to use the QL and an analogue interface to link to an x/y frequency plotter. Other uses include temperature measurement - taking direct readings off many temperature probes, with only a resistor network added to give the appropriate voltage range. TF Services use rheostats to provide position readings when running their "intelligent" robot demonstrations. Simon Goodwin is investigating sound sampling and playback. (Because of speed on the I2C bus, music will never be very high quality, but acceptable speech should be possible. It could be used as the basis or a computer-based multimeter, low frequency meter/low frequency oscilloscope.

The **PARALLEL INTERFACE** provides 16 two-way lines to read binary data (high/low). Output is high enough to drive LEDs, but has to be stepped up to drive relays/motors and so on. It can also be used for direct input to many digital model train controllers.

Future parallel circuit developments include a "power driver" to switch up to four amps. This is mainly to drive up to eight motors unidirectionally, but can also link into an 8-relay device (also on the drawing board) for mains or low-voltage switching.

The Parallel interface is priced £25, and the analogue interface £30, from **TF Services, Holly Corner, Priory Road, Chavey Down, Ascot, Berks SL5 8RL**. Note TF's new address! **Tel. 0344 890986. Fax** and scrolling modem **0344 890987.**

I2C data sheets, and control software and manuals can be bought for £2 an item without the interfaces.

Front End From Hessler

Albin Hessler has produced a new desktop front end for Qdos-compatible systems. Billed as "the ultimate pointer-driven desktop program for Qdos-compatible systems", **CueShell** has freely-sizable windows, drag-and-drop graphics copying, pan/scroll/slide bars, system/jobs/hotkeys control, saving of window sizes and catalogue sort orders, supports all screen resolutions and is easy to configure and put into use. (Albin Hessler has specialised in making processes easier in the past.)

CueShell can be used for file management, or for file-name selection from other programs. It requires expanded memory, and Albin recommends use of a mouse. Upgrades will be available for the cost of postage in International Reply Coupons.

The program will cost about DM100. Enquiries to **Albin Hessler Software, Im Zeilfeld 25, D-72631 Aichatal, Germany. Tel (and fax) 010 49 7127 56280.**

International Meeting in February

Sinclair QL User Club eV are to hold an International QL Meeting in Germany on Saturday 19th February 1994.

The meeting will be held at the University of Bielefeld in North East Germany, not very far from Munster where International QL Meetings have been held before.

Franz Herrmann has prepared a full pack of information including street maps, train connections, accommodation and sightseeing.

The address to contact is **Sinclair QL User Club eV, c/o Franz Herrmann, Talstrasse 21, D-53545 Ockenfels, Germany, or via Internet: Franz_Herrmann @ bn.maus.de.** (No unencoded material, no large texts please.)

See Franz's letter in **Open Channel** this month. Also reported to be happening some time next year (no dates fixed yet) are meetings arranged by Qltaly and Ergon Development in Italy, and IQLR in Newport, USA.

QUBBESOFT GO INTO DISK INTERFACES

Qubbesoft now expect to release their **Fastnet** QL networking interface at the end of January, priced around £120 for a two-board unit, which will link Ql to QL, ST to ST or QL to ST. The boards are with the pcb makers as we write.

Ron will be showing off Fastnet at Andrew Frannick's QL show at Blackrod, Bolton on 23rd January; Quanta's Edinburgh show on 12th February, and Franz Herrman's International QL Meeting in Bielefeld, Germany, on 19th February.

Qubbesoft are also launching a new QL hardware product shortly following the Fastnet. This is an **IDE hard disk interface, codenamed QUBIDE.** "Stuart thought up the name", says Ron. Miracle's Stuart Honeyball also thought of a way to connect the interface to the QL rom slot to plug-and-go, without an extra adaptor board. The Qubide may be ready by February 1994, at a price between £75 and £100. The closer they can get it to £75, says Ron, the happier they will be. The Qubide will drive one hard disk of up to 120 megabytes, which is plenty for nearly any application other than industrial cad or dtp, and more than enough for all the QL's memory-efficient applications. For more information contact **Qubbesoft on 0376 890986.**

MERZ'S NEW MULTITASKING CHESS

Jochen Merz Software have a brand new, multitasking chess program, **Black Knight**, for the Pointer Environment.

Black Knight runs with a 5000-move opening library, 10 levels of play from 5 seconds to 1 hour, and a chess clock. It can set up positions and load and save games.

Because the program will seize, in true battle fashion, almost as much memory as it can find to run the largest possible transposition tables, Black Knight benefits from hardware enhancements. However, it will run on a minimum of 640K memory, and will multitask with other programs (including itself) provided you tell it at the start how much memory you want to remain free.

The **Pointer Environment** (which is not supplied) must be installed before the game starts. Black Knight runs with the Gold Card, Atari STs with a QL emulator, and the QXL as well as other expansions with sufficient memory. TKII's default data directory can be used to load and save games if it is present, but is not necessary.

Both the mouse and cursor keys can be used, and Black Knight will disallow illegal moves.

The exclusive distributors of Black Knight are **Jochen Merz Software, Im Stillen Winkel 12, D-47169 Duisberg, Germany. Tel. +49 203 501274.**

QL World has a copy of Black Knight for review. Please contact the Editor.

Welsh Board Tackle Spelling!

The Welsh Language Board has come up with a Welsh Language spellchecker, a gargantuan task when one Welsh verb can produce 100 forms. At present, the spellchecker can only detect routine typos - full error detection can only be achieved when a grammar-checker is incorporated as well, for reasons which we're sure a Welsh speaker would like to explain! Unfortunately, the first release is only for WordPerfect 5.1 on you-know-what, but maybe some enterprising supplier could build the vocabulary into one of our QL word processors. Qwyll, for instance?

For more information contact **Huw Owen, The Welsh Language Board, Longcross Court, 47 Newport Road, Cardiff CF2 1AD. Tel. 0222 488745.**

10 YEARS OF DIGITAL PRECISION TOO!

Digital Precision are celebrating their **10th Anniversary** in 1984. Digital, the longest-lasting QL software company, launched their range after many months of preparation with three programs: QL Super Sprite Generator, QL Super Monitor (with free 68000 Disassembler) and QL Super Backgammon. It's fair to say that these products nicely represented the interest of Freddie Vachha, Digital Precision's substantial genius loci, in practical utilities and serious game programs.

QL Sprite Generator and Super Backgammon are still in the catalogue, and Super Monitor has been replaced by QMon Machine Code Monitor. Over the years these three have been joined by no fewer than 80 other programs, from Lightning (for many their first experience of a fast QL) to Perfect Pointer Tools, the Perfection wordprocessor to the Conqueror PC emulator, Professional Astrologer to Eye-Q graphics, Turbo Basic Compiler to Micro Bridge.

Digital's first advertisement naturally went with a splash, a full-colour page with discounts for buyers of two or three programs, and an appeal for good original QL software. And there for the first time was the distinctive Digital Precision logo.

Extraordinary as it may seem now, in the early days Digital Precision proceeded gradually, advertising occasionally and building their list slowly. By the end of 1986, however, they had transferred from Manor Road to 222 The Avenue, where they still are today, and added Professional and Super Astrologer, Eye-Q, Supercharge, Super Media Manager, Superforth and several games. The rest, as they say, is history!

Scan Digital Precision's software menu on **pages 2 and 3.**

Open Channel

Open Channel is where you have the opportunity to voice your opinions in Sinclair QL World. Whether you want to ask for help with a technical problem, provide somebody with an answer, or just sound off about something which bothers you, write to:

Open Channel, QL World,
The Blue Barn, Tew Lane,
Wootton, Woodstock OX7
1HA.

Non-Hexed

For people who don't want to try changing hex bytes to fix the Abacus AMEND bug in West Midlands Xchange, as described by Ron Stewart in QL World II.9, I am sending a listing that will fix it.

Howard Clase
St. Johns
Newfoundland
Canada

```
5 REMark A program to fix a bug in the West
10 REMark Midlands PD version of Xchange that
15 REMark causes a crash when the ABACUS "amend"
20 REMark function is used on a Gold Card.
25 REMark See QL World 1993 Vol2 #9 p 10
30 REMark H.J. Clase 1993.11.12
100 REMark ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^main
105 CLS: PRINT"Put your copy of Xchange into"
110 PRINT"flp1 and press any key when ready."
115 PAUSE (-1): OPEN#3,flp1_Xchange:
120 ad=41194: long_word$=Check_long_word$(ad)
125 IF long_word$="52390000"
130   PRINT "Bug found": BPUT#3\(ad),6,45,0,1
135   IF Check_long_word$(ad)="062D0001"
140     PRINT "Bug now fixed": END IF
145 ELSE IF long_word$="062D0001"
150      PRINT "Bug already fixed"
155      ELSE : PRINT "Not West Midlands version"
160 END IF : END IF
165 CLOSE#3
200 REMark ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
205 DEFine FuNction           Check_long_word$(ad)
210 LOCal i,b$,a%: b$=""
215 FOR i=ad TO ad+3:BGET#3\i,a%: b$=b$&HEX$(a%,8)
220 RETurn b$: END DEFine
300 REMark ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^End
```

PS - Budgie?! That's a common or waterside Kingfisher! What did you get in nature study?

Mostly disembodied bulls' eyes. The birdie is in fact Alcedo Atthis Barkerii, whose characteristics are a short beak and a garish habitat. It goes around in pairs, and lives on those little fish sometimes found swimming in monitors, and any bugs it can find. Every programmer should have one!

Downloadable Dumfounding

I would like to thank you and your team for a fine magazine. The articles are always meaningful to most of us ... have you ever read the average photo or radio magazine, or ...?

But I have some comments and hopes for the future. My set-up consists of a boxed QL with Gold Card and Minerva, with two 720K and two ED drives, external keyboard and the Mersey Mouse. The printer is a NEC Pinwriter P2200.

I have managed to get Wordperfect operational using the four drives (flp4_ as a pseudo hard disk), but I still prefer Perfection SE and to use the QL as a QL!

Now for some time I have envied other people with their more modern printers having available the **extra** fonts built in. Being one of the thousands thrown onto the heap by the march of time and the latest technological innovations, I can't go out buying the latest gear or expensive programs. I've played around with some of the relevant progs from the Quanta library, and was impressed by the Borman_Fonts.

But it's obviously not as simple as that! having become a little disillusioned and not a little confused about the whole thing, not getting the results I wanted, I wondered if there's a chance that QL World might have a future series, including answering the following:

Just what is the nature of built-in fonts in a printer?

Do downloaded fonts to the printer look the same to the Ser port as inbuilt ones?

Can these be used in the same way (say, from Perfection) as the normal fonts, ie underlining and Bold?

Would it be feasible to use Olde English and the like in Perfection and get a printout in the same font?

Also, would it be possible to do a rehash (they must have appeared some time, somewhere) of listings to Search and Replace (strings on a disk file)? And disk comparison and file comparison type programs (right up Simon's street, perhaps).

Again, many thanks for what we are getting. Seasons greetings.

Bryan Orgar
Ashford
Kent

Moving fonts from one source to another, especially one computer to another, can raise unpredictable bugs. Sometimes one has to refer to the makers of both for help. Apart from the famous (old) QL overheat, I have seen more systems (all kinds) crashed by font anomalies than any other cause! PD fonts give the most trouble. Fonts look superficially simple, but they are complex shapes and can be difficult for the operating system to handle. Oddly, simple fonts seem to *give as much trouble as fancy ones.*

The office font enthusiast says: "A downloaded font will look just like an ordinary printer font. There will be a special control code to switch to the downloaded font, just as there is a control code to switch to the resident fonts.

On laser printers individual fonts are provided for bold, italic as well as standard. I don't know whether the bold and italic versions fonts for the NEC P2200 are stored in the printer as separate fonts, or whether the printer simply approximates bold and italic itself. It is best to talk to the vendors (in this case Digital Precision) about fonts your wordprocessor can use."

I will make enquiries about the other matters.

Calling QL from Germany

On behalf of all our members we would like to express our gratefulness to all Sinclair QL User Groups, Publications, Hardware Producers, Software Houses, Dealers and Distributors, for your continued support of the Sinclair QL and compatibles, with hardware add-ons, new software developments, new computers and especially all kinds of service.

The situation of the QL scene has dramatically changed in the last two years, and there is a certain ambiguity in this process.

On the one hand, the last two years have seen numerous new products, high quality software and several successors for the Sinclair QL are now being offered, and an excellent service on commercial and private basis, namely user groups, printed publications, and electronic networks.

On the other hand, the membership of user groups is slowly fading away, and despite all the modern technology available, many users are giving up the QLIt is our intention to bring these two sides together by giving the highly active part of the QL scene a chance to demonstrate their developments and services to the more passive users, and we want to show pure users that the operating systems QDOS and SMS2/SMSQ have a future.

We also intend to bring together all sorts of people interested in the QL for informal exchange, to inspire each other and direct help. Finally, we want to provide commercial and private support organisations with the possibility of demonstrating their products or services.

This is to be realised at an International QL Meeting in Germany on February 19th at the University of Bielefeld, north-east Germany.

It must be clearly pointed out that this meeting has no commercial background, and that it is not profit-orientated. We are not charging an entrance fee, and fees for commercial exhibitors will be quite low. If you wish for more information, feel free to contact us for the full meeting information.

Franz Herrmann
Sinclair QL User Club eV
Talstrasse 21
D-53545 Ockenfels
Germany

POKE Modification

I've taken an interest in Dr. Teply's letter. I have not really been surprised, because a QLCF member previously told me that the trick doesn't work on his Gold Card (recent version) too.

My mind is that the POKE, POKE_W and POKE_L keywords work fine on my Gold Card (2.26, yellow pcb) in their original form, and doesn't work in the manner with some others GC roms.

I guess that Miracle have modified, for security reasons, the behaviour of these keywords, in order that it may not be possible to amend QDOS routines. If so, they can't be blamed, because POKEing in this memory area is very risky, and a "bad" POKE can lead to the QL crashing.

Provided I've guessed right, the solution to this problem lies in the program:

```
100 base=ALCHP(98)
110 RESTORE
120 FOR offset=0 TO 96 STEP 2
130 READ mot$
140 POKE_W base+offset,HEX(mot$)
150 END FOR offset
160 SBYTES flp1_pokes_rsp,base,98
170 RECHP base
180 STOP
190 DATA "3078","0110","43FA","003C","4ED0","7800","6112","1881"
200 DATA "4E75","610A","3881","4E75","6104","2881","4E75","7801"
210 DATA "3078","0118","4E90","6614","5543","6610","2876","9800"
220 DATA "2236","9804","200C","C084","6602","4E75","584F","70F1"
230 DATA "4E75","0003","FFC6","0550","4F4B","4542","FFC6","0550"
240 DATA "4F4B","4557","FFC4","0550","4F4B","454C","0000","0000"
250 DATA "0000"
```

The program creates and saves a file which, once LRESPRed, makes the keywords POKEB, POKEW and POKEL ready for use. This code is almost the same as the one which stands in the JM rom.

Could Dr. Teply use POKEB, POKEW and POKEL in lieu of POKE, POKE_W and POKE_L: I think this trick will also (for the same reason) allow him to modify Font 2 of the QL character set.

Please, somebody, let me know if the "cursor colours" trick now works, and correct mistake(s) if you then intend to publish this other trick.

Bruno Coativ
Rennes
France

Editor's notebook

Our lead article this month is a review of the newest upgrade to the long established QL flight simulator, Flightdeck, by Bernard Denchfield of Deltasoft. Ray Dawson not only assesses the program, but offers a guided flight for the newcomer, complete with rotation and navigation.

Bryan Davies is 'on holiday', partly due to pressure of work and partly due to the fact that his post didn't get through before Christmas. One of our contributors measured a time lag of nine days for a first class packet from Gloucester to Bedford. No doubt others have been experiencing the same delays. Our 'late bug' actually helped us get some stuff delivered this month.

Some late news about the workshop meeting in Edinburgh on February 12th dropped onto our doormat this morning (we still haven't seen it in Quanta), so the best course for anyone in the locality who wants details in a hurry is to contact Dr. Alan Perberton at Napier University, Craig Lockhart, Collington Road, Edinburgh. Tel. 031 660 1826 (evg) 031 650 88920 (day).

Last but not least - Happy New Year to all our subscribers, readers, and helpers. Here's to 1994 and next Hogmanay.

FLIGHTDECK 1.03

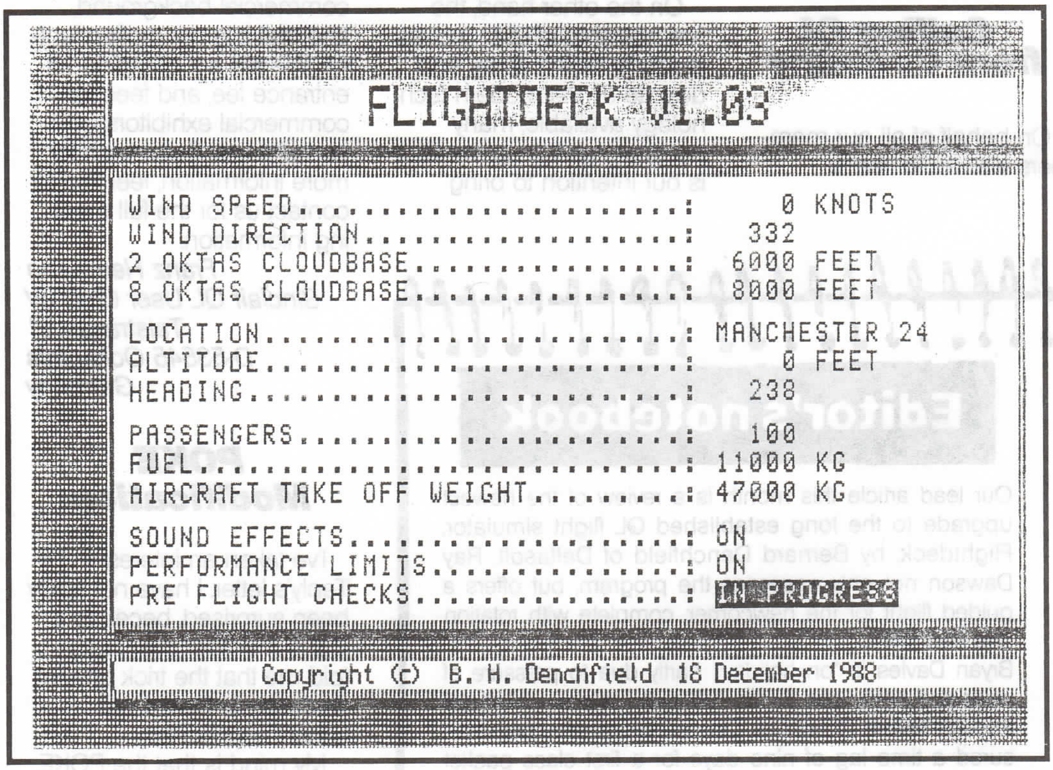
Ray Dawson takes off into the wild blue of Manchester.

INFORMATION
Program: Flightdeck V1.03
Publisher: Deltasoft via DJC,
41 Bro Emrys, Tal-Y-Bont,
Bangor, Gwynedd LL57 3YT.
Tel. 0248 354023.
Price: £15 cartridge or 3.5 in disk. Will run on unexpanded QLs.

Written mainly in machine code, Flightdeck allows you to "fly" a twin-engined jet airliner, and provides 3D views of the "world" outside and a database of 25 UK airports and over 200 navigation beacons. The plane has twin VHF omnidirectional range and distance measuring equipment (VOR/DME), automatic direction finding (ADF), and instrument landing system (ILS).

If you want to fly a light aircraft under the bridges of the River Thames, this program is not for you, but if you want to fly by instruments to major UK airports, this is your program. If you have watched the Boeing 737 flight simulator on The Krypton Factor on tv, then this is perhaps the easiest way to visualise Flightdeck. In the tv programme, contestants must land the aircraft using only instruments. Flightdeck's graphics are not in the same class as the 73 simulator, but a brave effort has been made by Bernard Denchfield to provide a meaningful view from the window, within the constraints of cost and graphics.

This view can be widened by 30 degrees in each direction by pressing the appropriate keys - just as if you were turning your head from side to side in a real cockpit. At startup, the pilot is provided with a screen to set up the parameters for the flight - see **Figure one** for the menu - using the up/down cursor keys and the left/right cursor keys. The flight is initiated by the space bar.



```
FLIGHTDECK V1.03

WIND SPEED..............................:      0 KNOTS
WIND DIRECTION..........................:    332
2 OKTAS CLOUDBASE.......................:   6000 FEET
8 OKTAS CLOUDBASE.......................:   8000 FEET

LOCATION................................: MANCHESTER 24
ALTITUDE................................:      0 FEET
HEADING.................................:    238

PASSENGERS..............................:    100
FUEL....................................:  11000 KG
AIRCRAFT TAKE OFF WEIGHT................:  47000 KG

SOUND EFFECTS...........................: ON
PERFORMANCE LIMITS......................: ON
PRE FLIGHT CHECKS.......................: IN PROGRESS

Copyright (c) B.H.Denchfield 18 December 1988
```

The selectable parameters are windspeed, wind direction, 2 octas and 8 octas cloudbase (more of this later), location, altitude, heading, number of passenger, amount of fuel, aircraft takeoff weight, sound effects (on/off), and performance limits (on/off). This latter option provides for future expansion, but I cannot think what form this would take! Pre-flight checks are last.

Octas and Cloud

The 2- and 8-octa cloudbases refer to the cloud cover in "one eighth of the sky" increments. One octa is one-eighth cloud cover, 2 octas is two-eighths, and so on. 8 octas means complete cloud cover. Setting the 2 octas level at 6000 ft, say, means that above 6000 ft one would expect to be in cloud 25% of the time, the cloud gradually increasing until 100% cloud cover would be encountered at, say, 8000 ft, if that value had been chosen as the 8 octas cloudbase setting.

The start location must be selected from the database by cycling through with the left-right cursor keys. Some 25 UK airports are included. Different runways may also be selected. For instance, "Manchester 24" is the main runway at Manchester Ringway when looking in a westerly direction - the 24 is the magnetic compass bearing of the runway, cut to two figures and rounded to the nearest 10 degrees. The actual heading of this runway is 238 degrees. The reciprocal bearing is 058 degrees, ie "Manchester 06", - the same runway from the opposite direction! When a landing is made on the ILS, the instrument displays the magnetic bearing 238 at the top as the selected Radial Beam, and its reciprocal 058 at the bottom to verify this.

The runway's magnetic heading is essential for a correct and successful approach and landing.

I found the starting altitude particularly useful, especially to avoid doing "take-offs" when practising landing, and you will need a lot of practice before you can execute a perfect landing. This option puts you at a pre-determined height above the airfield and on a pre-selected heading. Then you fly off for about ten miles, do a 180-degree turn and, having set the correct instrument Landing System frequency, NAV1, make an approach and landing, using the ILS

Instrument Landing System.

The sound effects can get a bit wearing, but I found them useful initially, as the engine noise saves looking continually at the throttle indicators to establish the thrust from each engine and adds a sense of reality. The squeal as the tyres touch down also adds reality on lading. The alternative - if you've made a hash of the landing - is a pair of black "curtains" closing across the window, with by a cryptic message such as "Crash - Undercarriage Not Down" or "Crash - Nose Wheel Collapse"! Fortunately, on a simulator one can always press the "Angel switch" and put the aircraft safely at the beginning of the simulation.

Manchester 24

A glance at **Figure two** will give you some idea of the Instrument Panel and the view of "Manchester 24" runway before takeoff. The panel is very well laid out, with 8 main dials. Engine, flap and landing gear indicators are in a "box" at the upper right of the panel. The navigation aid frequencies, NAV1 and VAV2 plus the DME1 and DME2 readings of the distance measuring equipment, along with ADF1 automatic direction finding beacon frequency, and the Fuel Indicator, are in another "box" at the bottom right. It is the VHF omnidirectional range, VOR1 and VOR2 frequencies, which are displayed as NAV1 and NAV2 in this box. DME1 and DME2 indicate the distances from those beacons. This enables you to get an accurate fix of your position, providing the aircraft is within range of the right beacons. If it is out of range, the VOR instruments display a red light to show that they are inoperative. The top left dial is the air speed indicator (ASI), calibrated in knots and with a digital readout at its centre. This measures the speed through the air (as opposed to speed over the ground) - there is a difference due to wind.

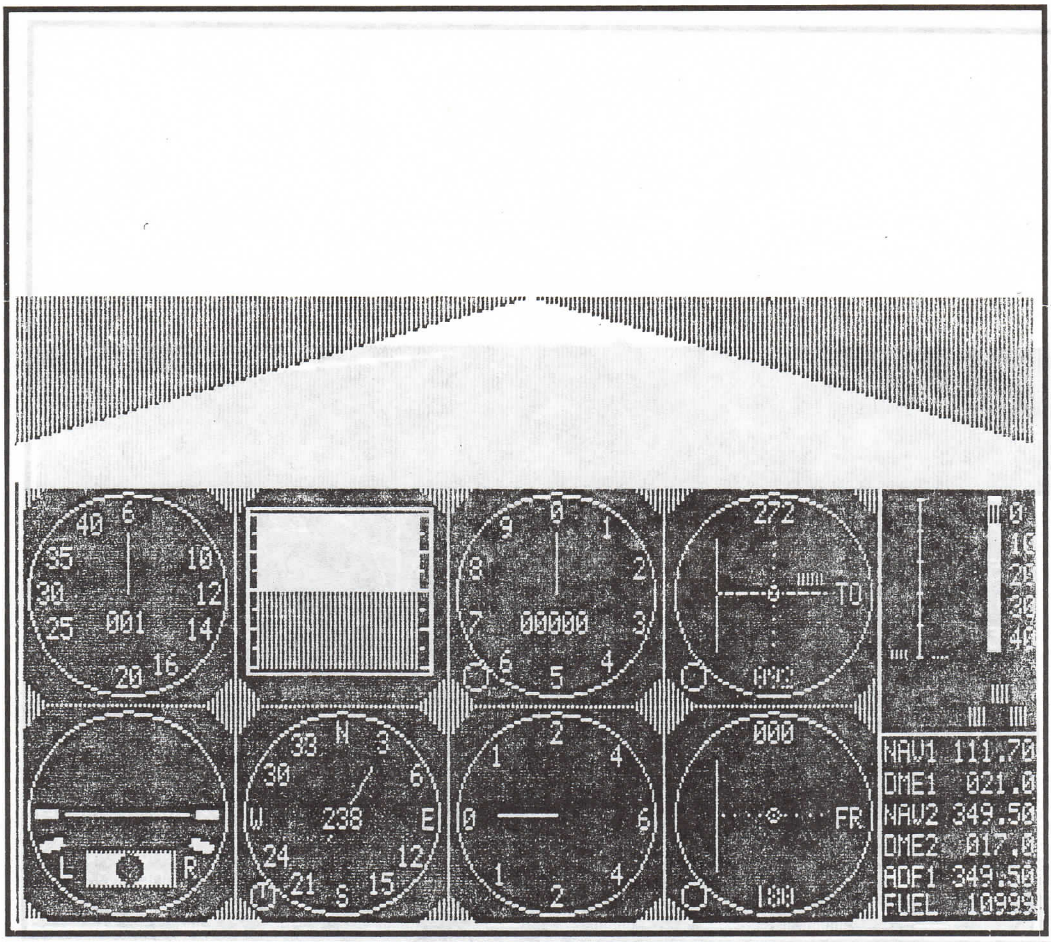
The artificial horizon is a square window with calibrated scales down each side. A shaded area in the square gives the aircraft's altitude. The larger scale markings represent 10 degrees of inclination, the smaller divisions represent 5 degrees within a maximum of plus and minus 20 degrees. If you can imagine looking straight ahead through a square window in the cockpit : in level flight, the horizon would appear to be across the middle of the window and horizontal. If you are flying "right way up", then the ground appears below the horizon and the sky above. The ground is shaded green and the sky white. If the aircraft is "nose up", the horizon moves upwards towards the top of the window. If the aircraft banks, the artificial horizon tilts, the degree of bank being shown on the scales at either side.

Next comes the Altimeter, calibrated in feet with a digital readout at its centre. It gives the aircraft's height above ground. (All airfields are assumed to be at sea level. This eases the burden of adjusting the altimeter for each airfield's height above sea level.)

On the extreme right is the VOR1 indicator (VHF omnidirectional range number 1) which also doubles as the instrument landing system (ILS) indicator, and this will be covered later.

The bottom left hand instrument is the turn coordinator. This indicates the rate of turn by means of a tilting horizontal line, similar to the artificial horizon. This line represents the aircraft's wings. The instrument has calibrated marks to indicate a Rate 1 turn, that is to say, a turn of 3 degrees per second change in the aircraft's heading. However, this instrument does not indicate the angle of bank, which is handled by the artificial horizon. Incorporated in this instrument is the balance indicator, which is used to determine if the aircraft is "skidding" or "slipping". This instrument contains a ball in a tube, which is displaced sideways by a sideways force, as may occur by applying rudder without an appropriate degree of bank, causing the aircraft to "yaw" or go into a flat turn, a most uncomfortable feeling. This rarely happens with this aircraft, as all turns are co-ordinated using aileron only, the correct amount of rudder being applied automatically.

The effect of skidding or slipping can only be experienced by using the rudder alone, causing it to skid, or by applying too much bank for the aircraft's speed to support, causing it to slip and possibly stall. There is an audible stall warning device. If this is ignored, then a "stick pusher" automatically prevents a stall by push-


NAV1 111.70
DME1 021.0
NAV2 349.50
DME2 017.0
ADF1 349.50
FUEL 10999

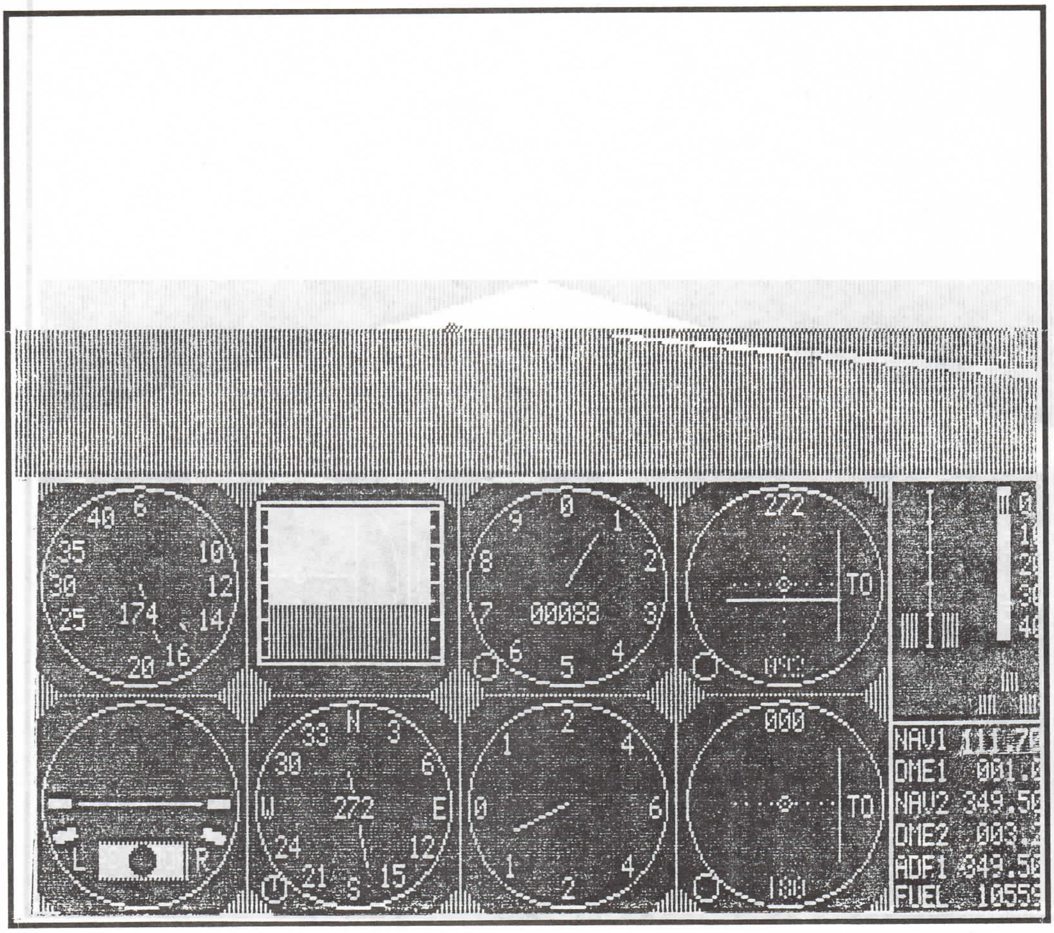
ing the stick forward to put the nose down. The air speed then increases and prevents the stall. You can still crash, though!

Next comes the automatic direction-finding (ADF) and compass. The compass requires little further comment, but the ADF beacons do. Each beacon transmits a signal on its specified frequency, at an equal strength to all points of the compass. The ADF needle in the aircraft always points to the beacon. It points vertically upwards when flying directly towards a selected non-directional beacon. If the needle moves to the right (clockwise), then the aircraft must be turned to the right to bring it back onto the correct heading., and conversely for the left. The aircraft's magnetic compass heading is displayed at the centre of this instrument. When an ADF beacon is overflown, the needle swings through 180 degrees and points directly downwards, as the beacon is now behind you.

Climb and Descent

Next is the vertical speed indicator (VSI). This indicates the rate of climb or descent, and is calibrated in thousands of feet per minute. When the needle is above the horizontal, the aircraft is climbing, and when below, it is descending, the rate of climb or dive being indicated on the marked space.

The bottom right hand instrument is the VOR2 (VHF Omnidirectional Range Number 2), and is similar to VOR1, except that it does not have the additional ILS facility. These beacons send "radial" signals just like the spokes of a wheel. Each beam is separated by 1 degree so there are 360 of them in all. By adjustment of the VOR, you can identify which radial the aircraft is crossing and turn on to that radial and fly towards, or away from, the beacon. The VOR/ILS beacon for each airfield is on the extended centre line of the runway so that flying along this radial, towards the airfield ensures the aircraft is on the correct heading. The ILS beacon also has a vertical fan shaped set of beams, to enable the aircraft's height, relative to its distance from the airfield, to be monitored. One of this set of beams enables a 3-degree "glide path" to be defined to assist a perfect approach and landing. Deltasoft are to be congratulated for a smart bit of programming here.



Perhaps the easiest way of learning about this is to do it.. Climb aboard! Strap yourself into the right hand seat. I will be in the left hand seat as the Captain, and we will go from Manchester to Liverpool. You have already guessed that I come from the North by my accent, and I have selected two northern airports, but the same rules apply when flying a short hop, or a much longer flight from Bournemouth to Aberdeen. It is much more realistic if you have got an air map of the terrain. If not, a road map will do.

Take a Map

Draw a pencil line on your map from the start point to your destination, and measure its true bearing with a protractor. Convert this to a magnetic bearing by adding the magnetic variation, currently about 6 degrees west, and so arrive at the compass "course to steer". Look along this line and establish which ADF or VOR beacons you need. If you're clever enough to be able to navigate in windy conditions, set up a wind speed and direction from the opening menu. Make the corrections to your heading if you do this.

We will assume no wind and a clear sky and set the 2 octas cloudbase at 6000 ft and 8 octas at 8000ft. We will fly at 4000 ft, so the trip will be cloudless. Set the location to "Manchester 24", altitude to 0 ft, and the heading will automatically set to 238 degrees. Leave the passengers, fuel and takeoff weight as default, and move the cursor to preflight checks.

Look up the frequency for Liverpool 27 ILS beacon - Ident ILQ - and get ready to set it on the ILS instrument, VOR1. Having pressed Space to start the flight and entered these frequencies as NAV1 and NAV2, check the Idents to ensure you are correct. Setting ADF1 to the same frequency as VOR2 may help navigation.

Simulators Only!

If you do any real flying, on no account must you use any of the VOR, ILS or ADF frequencies listed, or anything else in this program, while actually in the air. These frequencies are for simulator use only - you have been warned. Do a last check, take a look around to see the sky and runway are clear, select 10 degrees takeoff flap, open the throttles together slowly and evenly to full thrust.

As the aircraft starts to move, keep straight and level, using the rudder, if necessary, watching the airspeed indicator. With 100 passengers on board and 11000 kg of fuel, the "rotate" speed (take-off speed) will be about 150 knots. As this approaches, pull backwards steadily on the control column and the aircraft will gently become "unstuck", and climb. Hold this position until the artificial horizon indicates about 15 degrees of climb and then move the stick forward to maintain this attitude. Check the climb-and-dive instrument for a positive rate of climb, and raise the landing gear, checking that the gear warning lights go red toconfirm this. As the speed passes 240 knots, raise the flaps.

While climbing, ease back the throttles to about 70% thrust and turn the aircraft to its correct heading as per your flight plan, in conjunction with the navigation instruments. In a short time, VOR1 and VOR2 will indicate that they are operational (the red warning lights will go off) and the DME (distance measuring equipment) on VORs 1 and 2 will show the distance to your destination.These are not the same because one is the ILS beacon and the other is the VOR, a few miles apart - the ILS beacon is lined up on the runway and the other some distance away in the fields. Level off at about 4000 ft.

Try to check which "radial" you are crossing via the VOR instruments. It is important to get the final turn on the correct heading for your approach. In a big aircraft the controls will take effect slowly due to inertia. About 10 miles from Liverpool start descent by slightly reducing thrust and lowering the nose gently. Balance a reasonable rate of descent against throttle settings. When you are below about 235 knots, lower flaps and undercarriage, checking the warning lights. Raise the nose slightly, as increased draft from the flaps and undercarriage will enable an even descent with the nose about 3 degrees up. Adjust power and pitch attitude to keep the cross wires on the ILS aligned on the graticule. If the vertical wire is to the left of the centre line, you must do a gentle left turn to correct it. Do not chase the instrument backwards and forwards - make small corrections and wait for them to take effect. You should now see the runway, but watch your instruments. The ILS will assist you with correct height and heading, but you must control the airspeed. The landing speed is about 150 knots, so take this into account as you approach the runway. Fine adjustments to the controls will bring the aircraft safely to the threshold of Runway 27. "Flair out" by raising the nose slightly, then close the throttles and let the plane sink gently onto its main wheels. When you hear the wheels touch down, ease the stick forward to bring the nose wheel down and gently apply the brakes.

Well! Height not too bad, but the ILS needle says "fly right", and you ignored it. I could go on but this is your first attempt. Many enjoyable hours are yours, but be warned - it's addictive.

The more technically minded can extend the database of airfields and beacons. A full description is supplied. If I ever manage to execute two consecutive perfect landings, I may be tempted to try a few changes but, up to now, I have been fully occupied. I cannot resist the temptation, so here goes ... Happy landings, folks! Have fun.

Keyboard-90 Interface

Alex Munden offers a user report on his "Keyboard for the '90s".

INFORMATION

Manufacturer/supplier: Jurgen Falkenberg Computer Technik, Thanweg 361D-7539 Ersingen, Germany. Tel/fax: 010 49 7231 81058

UK Supplier: W N Richardson & Co (EEC), 8-21 Misbourne House, Chiltern Hill, Chalfont St Peter, SL9 9UE. Tel: 0753 888866. Fax: 0753 887149.

UK Price: Interface only £75.00 + p&p, keyboard only £30.00 + p&p, Interface and keyboard £95.00 + p&p, External fitting kit £14.00 + p&p.

Throughout QL history, a number of replacement keyboard systems have appeared but, for various reasons, have fallen by the wayside. Now Jurgen Falkenberg's Keyboard-90 interface allows any AT- or XT-style PC keyboard to be connected to the QL.

No nightmare

At first sight, it looks like an electronics novice's nightmare, comprising a 120x55 mm fibreglass printed circuit board, ten ICs, a handful of resistors, diodes and capacitors, a crystal, a DIP switch and an empty 40-pin IC socket. Also attached to the board is a length of cable and an in-line DIN socket. The amount of work required to convince the QL that it is still connected to its native keyboard can be seen in the form of a 32K eprom labelled QL-T90, which contains the firmware, and an Intel 8039 co-processor. The remainder of the chips are electronic latches and switches.

Internal installation of the interface may look like a major obstacle to anyone who has never seen the inside of their favourite black box, but things are not as complicated as they seem. What is mainly required is confidence and patience. The interface board displaces the Intel 8049 chip, the QL's door to the outside world. This explains the empty socket on the interface board. Removal is best done using an IC extraction tool or a flat bladed screwdriver. IC pins are quite soft, and can snap off very easily if bent or displaced.

You are then left with the problem of what to do with the DIN socket and its trailing lead. In my case, I found that, with a bit of persuasion, it could be eased into the slot in the casing alongside the tv modulator socket.

Then comes the moment to plug your PC-type keyboard into the DIN socket and switch on. If everything is working correctly, the normal start-up prompts should appear on screen. If not, press the reset button and try again. Your QL should respond to the new keyboard as though nothing had happened. If the QL still sulks, something is obviously wrong. Before you go any further, check that the pins on the interface board and the displaced 8049 chip are correctly aligned in their sockets. An error here is the most likely cause of failure.

Complete

Now for the good news. For anyone who has just read this with a sinking heart, a case and lead for external fitting of the interface are currently available from the UK supplier.

Anyone who has ever encountered a PC keyboard may be wondering how it relates to a QL. By now you should be familiar with the standard Qwerty pattern, but several key positions differ from the QL layout. If you purchase your keyboard with the interface, this should not be a problem, as new keytops will already have been fitted. If not, it's worth pointing out that the "@", "#" and copyright symbols are where you would expect to find them, but the pound sign sometimes nestles alongside the Enter key. The cursor keys are grouped together to the left of the main keyboard, and there are two extra Control and Alt keys, on opposite sides of the keyboard. They just duplicate each other (like the pair of Shift keys.)

All the extras!

Twelve function keys form a row along the top of the keyboard. F1 to F5 behave as you would expect, and F6 to F10 are coded as Shift-F1 to Shift-F5. F11 and F12 are the equivalents of Ctrl-F1 and Ctrl-F2 and could easily be detected by software. Also in this row are the ESCape key, and three keys marked respectively Print Screen/SYSRQ, which generates Ctrl-C; Scroll Lock, which returns Ctrl-F5 and Pause/Break, which is the equivalent of the QL's Ctrl-Space. To the right of the number keys is a back-arrow key, which duplicates Ctrl-left cursor as a delete-left key.

To the right of the main keyboard is a group of six keys marked INS, Home, PG Up, Del, End and PG Down, or, if you are lucky, more coherent equivalents. INS is the equivalent of Alt-Enter, Del replaces Ctrl-right cursor as the delete-right combination and Home, End, PG Up and PG Down are the respective equivalents of Alt-left cursor, right cursor, up cursor and down cursor. These have little bearing on Psion software but many newer programs use these combinations for screen navigation.

Finally, to the far right of the keyboard, is a 17-key numeric keypad. When Num Lock is pressed, an indicator led lights up, and the pad accepts numerical input from 0 to 9, plus the decimal point, the mathematical symbols and Enter. With Num Lock switched off, these keys duplicate the cursor and Alt-cursor keys, together with INS and Del.

Advantages

So, what's so special about attaching a PC keyboard to your QL? To begin with, there's the quantum leap from 65 to 102 keys. The keys themselves are properly angled for more comfortable input. Anyone who has spent hours slaving over a QL keyboard will know the strain it can place on an accomplished touch-typist, let alone the average "hunt and peck" operator. Keystrokes are more positive and give a greater degree of feedback, especially for non-typists who spend most of their time looking for keys, rather than confirming that key presses have been accepted on screen. The Caps Lock indicator light is a great improvement, particularly for those of us who still use Psion software. The single key replacements for many multiple keystrokes are a real boon and the separate numeric keypad really speeds up repetitive input, even for the 10% of the population who are "dextrally challenged". (Left handed, for those who did not realise that PC no longer means personal computer!)

All in all, adding a "proper" keyboard to a QL gives it a new lease of life. It's a bit like getting to know the QL all over again, now that you have all the benefits of Qdos and real multi-tasking, coupled with the advantage of a keyboard that does what you ask it.

The NEW USER GUIDE

Concepts Section

Section Thirty One

At least five separate layers exist between the CPU and the user.

Most of the difficulty in understanding computer concepts is that there are so many contradictory ideas to reconcile. Just when you have learned that microdrives are called mdv1_ and mdv2_, someone tells you that they can create the equivalent of a microdrive in the QL's memory, and call it mdv1_ or mdv2_ if they like. Even simple words like "memory" can be confusing: do I mean internal random access memory or external storage capacity? Why am I calling a microdrive an external storage device when it's located inside the QL's casing? A big help in understanding computers is to realise that they experience reality on several different layers, as if they lived in more than the three dimensions that we inhabit. Computers are incredibly stupid machines with a limited alphabet of two characters, a minimalist counting system of two digits and a philosophy of life that extends as far as the concepts of "Yes" and "No". To make computers look intelligent, software designers have to build up a set of codes out of strings of binary numbers. These codes are used as a basis for other codes, upon which yet more codes are based. Something approaching English language and sophisticated intelligence emerges near the top of the hierarchy.

It is convenient to divide the QL's world into five layers. The first is the physical, the things that are brought to life by electricity. The second is the machine code, the region of wall-to-wall binary numbers. The third layer is the operating system, which manages the housekeeping tasks in the background. The fourth layer is the domain of the command line and the SuperBasic programming language. The final layer is whatever program you choose to run on your computer. The differences between each layer and its neighbours might not appear great, but the difference between a circuit board full of electrons and a chess game capable of beating all but the very best players is enormous.

The Physical Layer

The life blood of a computer is electrical power, so it is appropriate that we begin our journey with an electron's-eye-view of the QL Computer components mostly run at between 5 and 12 volts, so the power supply is stepped down to an acceptable voltage before being distributed across the QL's circuit board, also known as the motherboard. The chips scattered across the motherboard, whatever their function, contain many tiny circuits etched into silicon wafers, each circuit set to be either "on" or "off". These are the equivalent of transistors. Just to confuse, when a circuit holds a voltage it might be deemed to be off, and when it has no voltage it might be deemed to be on. On and off and "current" and "no current" are simply opposite states and the meaning applied to them is arbitrary. People usually find it easier to think of "on" and "off" representing the binary digits 1 and 0. This is the first level of abstraction. The amount of information held in one bit is far too small to be of value, so computers deal with bits in groups of eight, sixteen or thirty-two. The QL shunts around bits in groups of eight, called bytes.

The heart of the computer is the central processing unit, a single, large chip that conducts most of the data processing. It contains several small storage areas called registers, each large enough to hold 16 bits. In some operations, registers are paired to hold a continuous binary number of 32 digits (or 32 bits, or 32 electrical charges). Unfortunately, to transport numbers to other components the CPU must first split them back into 8-bit (one-byte) chunks, leading to the QL sometimes being called an 8/32bit machine.

An important attribute of the CPU is its speed. The shortest operation carried out within the CPU takes one "tick". Unfortunately, even a simple task such as placing one character on the screen takes scores of instructions, and the QL must repaint the whole screen 50 times every second and still find time between each screen refresh to process information. This is possible because the CPU runs at the improbably fast speed of eight and a half million ticks a second, or 8.5 megahertz (MHz). For comparison, the human heart works at just above 1 Hertz.

At 8.5 million instructions a second, the CPU is a voracious consumer of information. If instructions and data were stored on magnetic devices such as microdrives the CPU would spend most of its time waiting for information to arrive and very little time processing it. To speed things along, data and instructions are transferred from microdrives and diskettes into silicon chips designed especially to store data. These work at around the 80-nanosecond mark, which is just about fast enough for the CPU to keep itself occupied between screen refreshes.

Silicon memory chips come in two forms. Read only memory, or rom, has its contents burned permanently into its circuits so they can never be altered. Random access memory, or ram, contains circuits that can be switched on and off. The disadvantage of ram is that the circuits must be refreshed with a new electric charge fifty times every second otherwise all of the ones turn to zeroes and the memory becomes empty. It would be nice to have memory that could be switched at will but which retained its settings when the electricity was turned off, but such chips are slow and very expensive.

A CPU and fast memory are in themselves a complete computer, but channels are needed for information to be pumped into the system and for the results to be fed out again. Devices that perform these duties include the keyboard, microdrives, disk drives, monitors and printers. Another term for a device is "peripheral": the two words are used interchangeably.

THE MACHINE CODE LAYER

Immediately above the physical layer is the haunt of the machine code junkies, although strictly speaking they use a first-generation, low-level language called Assembler. Machine code is based entirely on numbers, with the position of the number in a series denoting whether it is to be interpreted as an instruction or a piece of data or a memory address. A typical (but ficticious) command might be "204 12 45 65", which might mean "load the contents of memory address 3117 into the first register of the CPU". Whole sequences of commands form programs, and there is no other way of getting the CPU to do anything useful. Numbers might be the stuff of life to computers, but they are difficult for programmers to handle as a language, so most programming at this level is done using a relatively simple program called an assembler (hence "assembly language"). Assembly language is based on a set of mnemonics representing the numeric commands, so that the example quoted above might instead read "ld 12 45 65". Not much of an improvement, but sufficient to make it practicable to write machine code programs.

Early computers had very few instructions, but instruction sets have grown dramatically since the development of the silicon chip. The QL's Motorola 68008 CPU has hundreds of instructions in its vocabulary. The latest trend in CPU design is to reduce the number of instructions available while at the same time accelerating the speed at which they are carried out. This is the philosophy behind Reduced Instruction Set CPUs, or RISCs.

Each CPU has its own unique set of instructions, which is why machine code is sometimes known as "native code". QL programs are written with the Motorola 68008 CPU in mind. Computers with other chips, such as the Intel 80x86 family used in IBM-compatibles, cannot make sense of 68008 machine code, which is one good reason why QL programs cannot run on an IBM pc. Programmers have developed programs that run constantly to give the impression to other programs that they are passing instructions to one type of CPU when in fact they are running on another. Such programs are called emulators. It is easier to write emulators for CPUs in the same family, such as the QL emulator available for Ataris. However, the need to have a constantly-running emulator slows down the computer's ability to carry out its instructions.

The simplest machine code programs carry on running until they run out of instructions, always assuming that the flow of instructions continues to make sense. This is known as linear flow. Because some instructions might need to be carried out repeatedly, and other sets of instructions might need to be carried out only in specific circumstances, machine code includes labels, jumps, loops and branches. A label is a named location in a program that the CPU can search for. Instructions can be issued to force the CPU to go to a label, from where linear processing resumes. A jump is an instruction for the CPU to skip a given number of commands forwards or backwards before resuming linear processing. Jumping and label-seeking instructions can be part of a conditional instruction that in English might resemble "If x = y then go to label Start".

If the CPU moves backwards through a program to reach a label, or if it receives an instruction to jump back a few steps, the result is a loop. If the CPU moves forwards through a program to reach a label or as a result of a jump instruction it has created a branch. All machine code programs comprise segments of linear flow broken up by loops and branches. SuperBasic might dress it up a bit, but every structure in its vocabulary is either a loop (for instance, FOR ... NEXT and REPeat ... END REPeat) or a branch (for instance, IF ... THEN and SELECT ON ...).

Machine code programs are very powerful. Tiny routines can be developed to accomplish quite sophisticated tasks, but they are also difficult to design and write. A simple error such as writing "552" when you really meant "522" can make nonsense of an otherwise correct program. The computer might respond by carrying out inappropriate commands or by waiting endlessly for an instruction that never comes. These are examples of "crashes" caused by "bugs". Debugging machine code programs, which simply means correcting the errors in them, often takes programmers longer than the task of writing the programs in the first place.

The very first programs were written directly in machine code, but very quickly assembler programs came along to simplify the job. Assembler is therefore known as a first generation computer language.

THE OPERATING SYSTEM LAYER

Operating systems are suites of utilities written in machine code to handle frequently-used routine tasks. This saves programmers the effort of producing their own utilities for each of their programs. Instead of writing a little program that causes dots in the shape of an "A" to appear on the screen, programmers simply pass an "A" to a pre-written routine that does all the rest of the work for them. These facilities are so time-saving that they are used by almost every machine code programmer, except on rare occasions such as when programmers got so fed up with the QL's screen-handling routines that they write speed-up programs to replace them. One side-effect of this reliance on the operating system is that programs tend to share a "personality" imposed partly by the instruction set recognised by the CPU and partly by the operating system utilities available.

The operating system's primary concern is communication with peripheral devices. The QL's Disk Operating System (or QDOS) handles all communications to and from microdrive cartridges, the keyboard, the printer and the monitor. Despite its name, QDOS does not include the code to handle disks! These are provided by QDOS extensions written by Tony Tebby.

The suite of small routines that make up an operating system works on a simple basis. Some scrap of information is put into a specific location in the computer's memory and then the appropriate utility is called to do something with it. The information might be the number code representing a capital "A", and the called routine might print it onto the screen. QDOS also needs to store banks of ever-changing information regarding the current display mode, the size and colour of each of the windows, the characters typed at the keyboard that it hasn't had chance to operate on yet, and so on. This data is stored in tight-knit areas of random access memory called "system tables". When you issue the command

```
PAPER #1,4
```

QDOS places the value "4" into the right address inside a system table so that next time it needs to refer to the background colour of the default window it remembers that you have asked for green. The system tables occupy a large area of random access memory that QDOS grabs for itself as soon as the QL is turned on.

THE COMMAND LINE LAYER

The command line layer, which I have extended to include normal SuperBasic programs, is the level QL owners perhaps know best, and needs least explanation. Although obvious to most users, it is worthwhile to state that SuperBasic is actually a machine code program that constantly monitors keyboard and file input. At the end of the boot-up sequence, QDOS begins running SuperBasic and allocates it the special program identification of 0,0.

SuperBasic is an interpreted language. In other words, the SuperBasic interpreter analyses directly and executes the program code that you write. In effect, a SuperBasic program forms a special type of data input for the SuperBasic interpreter program to operate on. Other programming languages, including the QL's Turbo, have their source programs compiled into faster machine code routines by a utility called (unsurprisingly) a compiler. Once in machine code they do not need the machine code SuperBasic interpreter to interpret them: they deal directly with the CPU and operate much faster as a result.

Unless you load a specific type of program, the QL's layers stop here with SuperBasic. However, command line interpreters are becoming very rare these days, and for some QL owners SuperBasic is rarely used, hence the inclusion of the final layer in this article.

THE EVENT-DRIVEN LAYER

Many people think it unfriendly for a computer to hang around waiting for them to type something and then refuse to co-operate because the spelling or the punctuation was not precisely up to scratch. Instead of giving a command to re-size a window, they would prefer to drag the window borders around the screen, or call up a menu option called "Load" which lists all the available programs, rather than type "LOAD flp1_quill" (for instance). These facilities are provided by event-driven interfaces, epitomised on the QL by QPAC II by Tony Tebby. QPAC II replaces command-line SuperBasic with menus, pick-and-point lists and dialogue messages. There is less to learn, because the computer prompts you with the available options. It is a safer environment because inappropriate actions are forbidden and potentially dangerous ones (such as file deletion) must be confirmed. Event-driven interfaces often work best with a mouse - a rolling device that controls a pointer that floats across the screen. When the pointer is over the option you want to carry out, press the button on the mouse instead of hitting keyboard Enter, and you have the equivalent of a SuperBasic command without touching the keyboard.

Computer games are almost all event-driven, often controlling many functions with just a few keys. In the early days of microcomputing, games programmers taught their mainframe colleagues valuable lessons in concise coding and rich data structures. Now games lead the way in offering friendly and powerful interfaces, so why not put similar interfaces onto business programs? The QL screen display, unfortunately, lacks detail and only has a few colours, but a range of workable interfaces is nevertheless now available.

The QL's main strength is its operating system, still the best available for any micro on the market. In turn, QDOS supports powerful programs that are very compact compared with their MS-DOS and Unix equivalents. It is a great shame that weak hardware and poor marketing spoiled it for Sinclair.

GLOSSARY

68008 The least powerful member of the Motorola 68000-series of CPUs. The 68008 shares the same instruction set as its siblings but only has an 8-bit data bus (which also provided the "8" in its name).

bug An error in any type of program that leads to the computer doing something wrong or not doing something right. Relatively harmless bugs are sometimes jovially called "features".

bus A means of channelling digital information (in the form of electrical signals) around a circuit board. In a computer, buses are wires or copper lines, usually in clumps of eight, sixteen or thirty-two which operate in parallel. The two most important buses are the data bus (which transports data) and the address bus (which simultaneously carries instructions about where to store the data).

crash When the computer stops responding to input. This might be due to a bug, a power fluctuation or a power failure. Because the computer has to be re-started, any information stored in ram and not saved in a permanent medium elsewhere will be lost.

cpu Central Processing Unit, the large microchip which contains the main processor or "brain" of the computer.

diskette (disk) A flat disc of video-quality magnetic tape in a case. Magnetised heads in disk drives write data onto diskettes by polarising small sections of the disk. Other heads can read the information back again.

file Any digital information stored under a single filename or header on a disk or microdrive cartridge. The location marker contained in the filename is the basic means of storing and retrieving computer information.

microdrive A means of reading microdrive cartridges. Microdrives contain a reading and a writing head, like a disk drive (see above).

microdrive cartridge (microcassette) A means of storing data magnetically: a loop of about 22 inches of video-quality tape inside a small cartridge.

nanosecond A thousand-millionth of a second, the standard unit for measuring short wavelengths in the class including visible light and electrical current.

ram Random access memory, called so because it takes the same amount of time to retrieve one byte of data no matter where in the ram it is stored. This is not true of tapes and disks because the heads or tape must first move to the right location.

rom Read only memory, called so because its contents can be read into the CPU, but any attempt to write data back to rom addresses fails.

*****

WHAT IS BINARY?

The decimal counting system, or base-10, comprises ten digits, 0 to 9. Provided we use enough of them, these can represent all the finite, real numbers that we can imagine. The binary counting system, or base-2, only has two digits, 0 to 1. Again, provided there are enough of them they can represent any finite, real number. The main advantage of binary is that its digits can be represented by the "on" and "off" states of electronic circuits. The main disadvantage is that you need so many of them to represent even quite modest numbers. Any value from 100 to 999 can be represented using three decimal digits, but three binary digits can only represent 100 to 111 (4 to 7 in decimal). A lesser disadvantage is that the "significant" numbers in the decimal and binary counting systems rarely co-incide. The concept of significance is entirely arbitrary and base-related: 1000 is only thought to be more significant than 986 because of its nice pattern of zeroes. Look at the stock market: if prices fall below a value that ends in lots of zeroes it makes the evening news. The binary value of 1000 is 1111101000, an entirely insignificant collection of digits. The decimal value 1024 is of more interest to computers because it is a one with ten zeroes behind it - which explains why a 128K computer has 131,072 bytes rather than 128,000 bytes in its memory.

One does not need to count in binary to use a computer, but significant binary values crop up all over the place: the memory expansion options, the number of colours that can be displayed, the lack of the flashing status in high resolution mode, the total number of characters in the Ascii character set. If you are a programmer you should at least recognise the significant binary values.

Section Thirty One

5th Italian show

Despite torrential floods and a national railway strike, QL enthusiasts from Italy and the UK flocked to Reggio Emilia for the 5th Italian QL Show this autumn.

The traders included Tony Firshman, Bill 'EEC Richardson, Ergon Development, Miracle Systems, SPEM, the QItaly user-group and Qubbesoft. The venue was a large room on the top floor of Reggio's modern Civic Centre, with that all-important en suite pizzeria. Admission was free to both traders and customers.

Roberto Orlandi and Eros Forenzi of QItaly were among the first to arrive, with an impressive A4 magazine QL Mondo, copies of their disk newsletter, QPac upgrades, QL books in Italian and many items of hardware and software for sale.

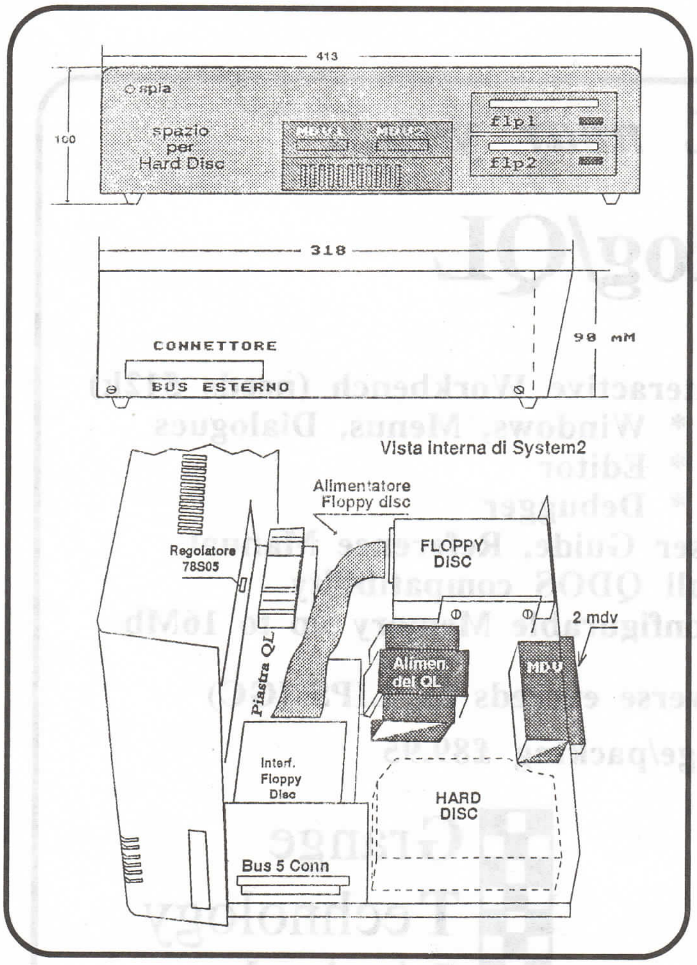
These two are Qdos fanatics, with 'QL' stickers on the backs of their cars which confuse lesser Italians, for whom 'QL' is an abbreviation for a fraction of a ton. When Computer Shopper had a QL column Eros used to drive 50 kilometres to Switzerland each month for his 88p copy!

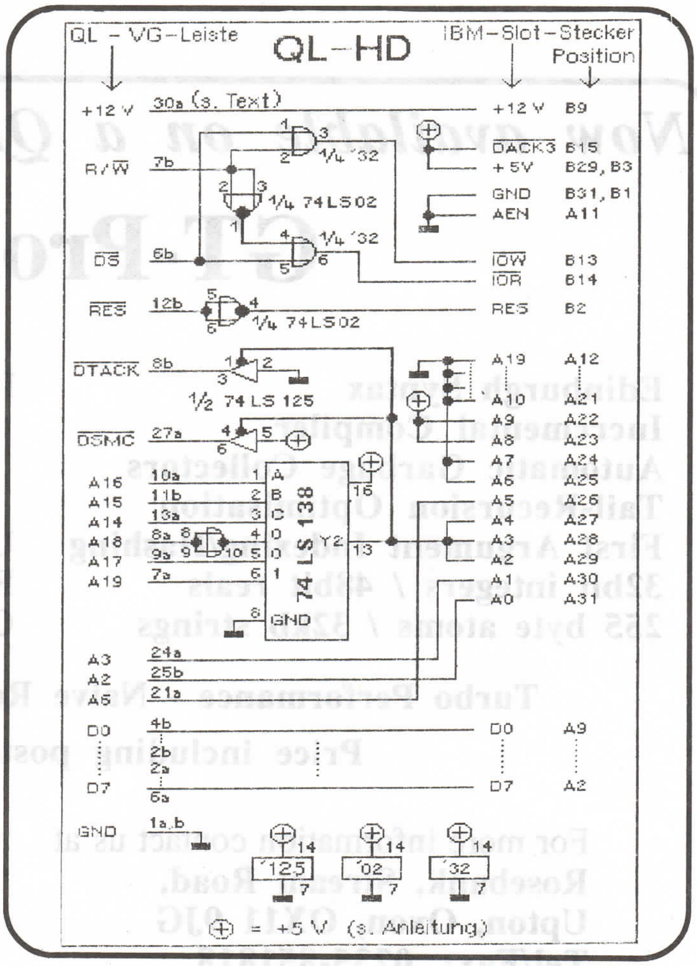
Since the demise of Sandy, SPEM have been kings of the Italian QL hardware scene. They were showing off their QL expansion keyboard, which has gained a new lease of life since the arrival of Hermes, and their impressive System2 QL expansion chassis.

This is a white metal box with a flip-top lid, with fittings for the QL circuit board, hard disk, two floppies, two microdrives, power supply and all your interfaces. With the exception of the microdrive extension port, all the Sinclair connectors remain available around the edge of the chassis.

System2 found users on several stands at the show, alongside custom variants, QXL-powered PCs, and original QLs. Even one shy Amiga 500 made an appearance. If you have no qualms about sawing your QL in half, to extract the microdrive mountings, the System2 is almost certainly the QL box for you. SPEM also sell a range of expansion boards and EPROM cartridges.

Ron Dunnett, Tony Firshman and Stuart Honeyball had planned to





travel by train and their customary folding bicycles. Alas Italian railway workers took Sunday off in protest at impending privatisation. The contingent arrived, sleepless but intact, after an eventful 17-hour drive from Paris, dodging storms, Alps and flooded motorways.

I arrived by plane and train a couple of days early, the guest of Ergon boss Davide Santachiara, and spent an entertaining weekend trying to coax secrets of the **ZM/HT compiler** from its enigmatic co-author Marco Ternelli. Ergon's best-seller at the show was their new Disk Utilities, capable of repairing Gold Card HD and ED disks, and formatting HD to over 1.52 megabytes. Gold Card and Spectrum fans upgraded to the new ZM/128 emulator.

Stuart Honeyball of Miracle Systems gave a talk on QXL and his plans for the future. The next release of QXL will include network and serial device support, with floppy-disk formatting not far away. The new SuperBasic interpreter has been delayed, because Laurence Reeves has broken his arm.

Work on the 'low-cost' QL SCSI interface is well under-way, but Stuart told the show that the QL graphics board is on the back burner till the QXL software is more complete. Meanwhile QXL users can expect regular software updates.

Tony Firshman demonstrated the first production peripherals for his **Minerva I2C interface.** These black boxes are labelled in red and festooned with D-type sockets. They are controlled by the same Philips circuits as Minerva's Mark 2 clock chip, and you can connect up to eight interfaces of each type to one QL. Large collections will need an external power-supply.

The eight-bit Analogue interface has eight inputs and two outputs. The simpler digital interface has sixteen five volt outputs which can be turned on or off from SuperBasic, or re-programmed as digital inputs. Hiigh power versions for AC and DC control are coming soon.

Bill Richardson, formerly of EEC Ltd, brought hardware and brochures from the UK and Germany, most notably including Jurgen Falkenburg's **QL hard disk adapter,** which uses a PC-XT interface and drives. This really needs a box, but should still find users as it's the only plug-and-go hard disk system currently on the QL market.

Impecunious enthusiasts may favour Dirk Steinkopf's **DIY approach,** which uses just four TTL chips, worth about a pound, to adapt the QL expansion connector signals to a simulated PC/XT slot. A full 68000-code device driver is on disk, with a circuit diagram, utility programs in C and documentation in German and English.

Dirk's kit has been tested with OMTI and Western Digital hard disk controllers, specified in the documentation, and suits both MFM and RLL drives. It is said to co-exist happily with Gold and Trump Cards and supports QJump level two devices, with subdirectories. My copy came from Ergon, and is freely distributable; you can get it from bulletin boards, user groups and PD libraries, such as **Qubbesoft's PD disk SPECIAL 22.**

Participants found it easy to communicate in a mixture of English, Italian, Latin and French, and it is a pity there were not more people from Northern Europe at the show. Perhaps storms and railways defeated them. On this showing, the QL is keenly and imaginatively supported in Italy; the next meeting should be just as rewarding, and hopefully a little sunnier.

DIY TOOLKIT

Simon Goodwin teaches the DIY Mouse to emulate keyboard entry for older programs.

The small MOUSE_PAINT listing is a simple mouse-driven painting package in SuperBasic, which uses the DIY Mouse extensions. It lets you move a cursor anywhere on the screen, leaving a trail of dots in one of the eight possible colours, depending on the combination of three buttons being pressed as it moves. You can view it as a sort of colour etch-a-sketch.

MOUSE_PAINT needs a three button mouse to allow access to the full range of Mode8 colours, but you can use it in four colours with a two-button Microsoft mouse.

A real mouse painting package would reserve certain areas of the screen for ink post and other controls, rather than dedicating the buttons to colour control, but the example does show how easily colour mouse drawing can be implemented.

The PD Qdos emulator for the Amiga includes another SuperBasic program that uses the PTR_ extensions. PALETTE_BAS, on the Qdos utilities disk, lets you choose the QL emulator screen colours from a palette of 4096 shades. You use the mouse to select and move sliders on the screen, and the colours change as you do it. Alas, this only works on machines with palette hardware.

Keyboard Emulation

It is always best to tailor programs specifically for the mouse, capable of tracking and responding to precise mouse movements and button combinations. Unfortunately most QL programs were written before there was a standard for SuperBasic mouse commands and functions, and it would be good to be able to use them with the mouse even though they were written for keyboard or perhaps joystick control.

That is the main reason for this update to the DIY Mouse project. The extra code in the main listing adds optional emulation of cursors and other major control keys via the mouse. Thus the mouse works with Quill, ED, Devpac, Turbo, Abacus, Spy, Editor, and other cursor-controlled packages.

First you must load and turn on the appropriate code, and set the right baud rate: normally BAUD 1200. The mouse functions will now respond to movements of the mouse or buttons, but keyboard entry will be unaffected. When you want keyboard emulation, so the cursor tracks mouse movements, issue the new command:

PTR_KEY 1,0

The mouse will now generate cursor key-presses as it moves. Use PTR_INC to set the speed, and PTR_KEY 0,0 to turn it off. You can still use the keyboard as normal. The mouse signals are in addition to joystick or key-presses.

Mouse Wrapping

The second parameter of PTR_KEY is another switch. It allows mouse co-ordinates to wrap from one edge of the screen tto the opposite edge.

```
* QL WORLD DIY TOOLKIT - MOUSE/POINTER DRIVER + KEY EMULATION
* Links SuperBASIC and Qdos key queues to a PC serial mouse
* Version 2.8 updates only, Copyright © 1993 Simon N Goodwin
*
* Extensions over version 1.n:
*
* 2.2 Optional co-ordinate wrap-around at screen borders
* 2.4 PTR_KEY extension to control wrap and key emulation
* 2.7 Optional Keyboard emulation for cursors and buttons
* 2.8 PTR_FN% function to read pointer characteristics
*
qkeys       equ     1               Configure to zero or one
*
** 2.0 ** Ensure QKEYS is valid (0 or 1); value is used later
*
            ifne    qkeys
            ifne    qkeys-1
            fail                    QKEYS 0 or 1 = key emulation
            endc
            endc
*
** 2.0 ** Key codes
*
left        equ     192             Cursor arrows
right       equ     200
up          equ     208
down        equ     216
button1     equ     32              Space
button2     equ     10              Enter
button3     equ     27              Escape
*
sv_keyq     equ     76              Key queue system variable
*
* These variables are relative to PREFIX
*
```

If you set PTR_KEY 0,1 before loading MOUSE_PAINT you will find that the cursor swaps round at the edges of the screen, giving you a quick route to the other side, top or bottom. This 'wrapping' is a popular option available on Apple Macs and other windowing systems. It was easy to do, so I added it for the second version.

PTR_KEY works like all the other commands to set variables, which is why I lazily implemented one command to control both the version 2 improvements - pointer wrap and keyboard emulation.

The parameters are simple switches, with zero meaning off, so PTR_KEY 1,0 turns cursor key emulation on and co-ordinate wrapping off. The order should be easy to remember once you note that the WRAP parameter is on the outside.

PTR_INC

PTR_INC sets the number of mouse pulses that correspond to one move of the cursor in either direction. It was introduced a couple of months ago, but finds its niche in the new version 2 of the DIY Mouse driver, which sends cursor-control characters to tasks normally controlled from the keyboard.

The increment is in Mode4 pixels, and starts off at 12 in X and 24 in Y. This is less sensitive than the settings for the first version, published in QL World 1993 issue 10, but it works better for me. It depends how far you want to move with one prod of the mouse.

If you use Mode8 or wide Mode4 characters the cursor will respond much faster to horizontal moves than to vertical ones, and it may be worth increasing the X increment, with a command like PTR_INC 24,24. It pays to experiment - some people like slower vertical movement since it reduces the risk of moving off the current line when scanning left and right.

You can use PTR_INC to scale the movement on screen to suit your mousepad. Staying in Mode8 for the time being, try PTR_INC 30,25 for easy character-positioning, and PTR_INC 6,5 if you want fast movement. Double the first value, the X increment, if using Mode4, and double the second value to suit double-height characters.

Main Listing

The main listing this month consists of updates to the DIY Mouse driver source for Version 2. This implements keyboard emulation, so that programs can be controlled with the mouse instead of joystick or cursor keys. The lines have been extracted from the full MOUSE_ASM source, which is available as usual to readers on disk or cartridge.

DIY Toolkit Volume I includes all the new source code, and eight ready-made mouse drivers. The volume is available by post from former Quanta editor Dr. Bill Fuggle. Volume I consists of SuperBasic prototypes, wiring details, example programs, assembler source and binary code for "Mouse Systems PC" and "Microsoft" serial mouse drivers to suit either serial port.

DIY Toolkit volumes cost three pounds each on disk or microdrive cartridge, anywhere in the world, and come with laser-printed documentation if you order two or more. Twenty four volumes are available from **DIY Toolkit, 86 Lordswood Road, Harborne, Birmingham B17 9BY.** Please make cheques payable to DIY Toolkit, and send a stamped self-addressed envelope if you would like further details.

If you really enjoy typing you can merge the new lines into the source listed in previous issues. The new lines are intended to be added to the first two parts of the listing, and come in three sections. The first section gives names to key-

```
*                             PTR_FN%
latest_x     equu     0         0        Current co-ordinates
latest_y     equ      2         1
limit_x      equ      4         2        Right margin limit
limit_y      equ      6         3        Top margin limit
step_x       equ      8         4        Counts per horizontal move
step_y       equ      10        5        Counts per vertical move
button_bits  equ      12        6        Bits shadow each button
synchro      equ      14        7        Input byte number, 1 to 3
initial      equ      15                 Two-button initial byte
serial_id    equ      16        8+9      Zero or serial channel ID
*
             ifne     qkeys
drift_x      equ      20        10       ** 2.0 ** Accumulated X drift
drift_y      equ      22        11       Accumulated Y drift for keys
key_flag     equ      24        12       Set to request key queueing
wrap_flag    equ      26        13       Set to allow wrap at edges
             endc
*
var_end      equ      20+qkeys*8         ** 2.0 ** 8 bytes key space
*
*****************************************************************
*
* TWO BUTTON KEYBOARD EMULATION & CO-ORDINATE WRAPPING UPDATES
*
* (1) Replace BPL.S with word BPL to BYTE_THREE
* (2) Insert BSR BUTTON_KEY before storing D0 at BUTTON_BITS(A4)
* (3) Revised X and Y move handlers:
*
            bsr       horizontal       New line to drive key queue
            move.w    latest_x(a4),d0
            add.w     d1,d0            As before, so far
            bpl.s     enough_x         Check the upper limit
            tst.w     wrap_flag(a4)    Are we wrapping?
            beq.s     clear_xout
            add.w     limit_x(a4),d0   Wrap co-ordinate up
in_xrange   move.w    d0,latest_x(a4)
near_xrange bra       next_byte        Local diversion
*
enough_x    move.w    limit_x(a4),d1
            cmp.w     d1,d0            Over the top?
            bls.s     in_xrange
            tst.w     wrap_flag(a4)    Are we wrapping?
            beq.s     clip_x
            sub.w     d1,d0            Wrap co-ordinate down
            bra.s     in_xrange
clip_x      move.w    d1,d0            Set current Y to limit
            bra.s     in_xrange
clear_xout  clr.w     d0               Zero
            bra.s     near_xrange      Loop via local label
*
```

```
* Corresponding changes to the code to process Y deltas
*
             bsr      vertical        Drive the keyboard
             move.w   latest_y(a4),d0
             add.w    d1,d0           D0 is new candidate Y
             bpl.s    enough_y        Check the upper limit
             tst.w    wrap_flag(a4)   Are we wrapping?
             beq.s    clear_yout
             add.w    limit_y(a4),d0
             bra.s    in_yrange
*
enough_y     move.w   limit_y(a4),d1
             cmp.w    d1,d0           Over the top?
             bls.s    in_yrange
             tst.w    wrap_flag(a4)   Are we wrapping?
             beq.s    clip_y
             sub.w    d1,d0           Wrap co-ordinate down
             bra.s    in_yrange
clip_y       move.w   d1,d0           Set current Y to limit
*
in_yrange    move.w   d0,latest_y(a4)
near_sync    bra      clear_sync      Get ready for new message
clear_yout   clr.w    latest_y(a4)
             bra.s    near_sync       Return via local label
*
****************************************************************
*
** 2.0 ** Cursor movement routines; add near end of listing
* On entry D1 contains the new move delta and is preserved
* Uses A4, A6; corrupts A5, A2, D0, D3, D7; IO.QIN hits A3
*
horizontal   tst.w    key_flag(a4)
             beq.s    no_keys         Key entry disabled
             move.l   sv_keyq(a6),d0
             beq.s    no_keys
             move.w   d1,d7           Save delta
             move.l   a2,d6           Save input queue pointer
             movea.l  d0,a2           A2 -> Key queue
             move.w   drift_x(a4),d3
             movea.w  $E0.w,a5        IO.QIN vector
             add.w    d1,d3           D3 is total delta
             bmi.s    go_left

             moveq    #right,d1
*
more         sub.w    step_x(a4),d3   Try a move
             bcs.s    add_xout
             jsr      (a5)            Cursor right
             bra.s    more
*
go_left      moveq    #left,d1
less         add.w    step_x(a4),d3   Try a move
             bpl.s    sub_xout
             jsr      (a5)            Cursor left
             bra.s    less
*
sub_xout     sub.w    step_x(a4),d3
             bra.s    set_xdrift      Store remainder
add_xout     add.w    step_x(a4),d3
set_xdrift   move.w   d3,drift_x(a4)
             move.l   d6,a2           Restore serial queue
             move.w   d7,d1           Restore delta
no_keys      rts
*
vertical     tst.w    key_flag(a4)
             beq.s    no_keys
             move.l   sv_keyq(a6),d0
             beq.s    no_keys         No queue to be stuffed
             move.w   d1,d7
*
             ifeq     buttons-3
             neg.w    d1              Reverse Y direction
             endc
*
             move.l   a2,d6           Save SER queue address
             move.l   d0,a2           A2 -> Key queue
             movea.w  $E0.w,a5        A5 = IO.QIN vector
             move.w   drift_y(a4),d3
             add.w    d1,d3           D3 is total delta
             bmi.s    go_up
             moveq    #down,d1
descend      sub.w    step_y(a4),d3   Try a move
             bcs.s    add_yout        Don't go too far
             jsr      (a5)
             bra.s    descend
```

board codes and variable offsets in the interrupt linkage.

These variables are private to the handler and not directly available to Basic, but there are lots of interesting things among the interrupt handler's variables. It seems a good idea to have some way to read them all, so I have added a routine to do this in the DIY Toolkit Volume. The assembly code is similar to that for BUTTON%, but the parameter is a word number to be used as an offset when reading.

PTR_FN% can read any word of the DIY Mouse variables. It takes one parameter, the offset of the required word, starting at zero (LATEST_X), so PTR_FN%(2) reads the right margin limit (the third word) PTR_FN%(4) reads STEP_X and so on. Parameters for PTR_FN% are tabulated near the start of the new assembler listing.

Conditional Code

The single file MOUSE_ASM can create eight different versions of the driver, so it uses conditional assembly. The lines between an IFEQ or IFNE and the matching ENDC are only included in the assembled code if the expression after the IF is zero (IFEQ) or non-zero (IFNE). If your assembler does not support conditional assembly you need to comment out the lines that you do not need.

Set QKEYS to one or zero depending on whether or not you want extra code to give the option of keyboard emulation. You do not need this for 'mouse aware' programs that use extensions like PTR_X% and BUTTON% to read the mouse directly and precisely.

If QKEYS is one the mouse driver has the ability to send cursor movement characters to the current task, so you can move the cursor with a mouse instead of the arrow keys. The sub-

```
*
go_up         moveq     #up,d1
ascend        add.w     step_y(a4),d3   Try a move
              bpl.s     sub_yout
              jsr       (a5)
              bra.s     ascend
*
sub_yout      sub.w     step_y(a4),d3
              bra.s     set_ydrift
add_yout      add.w     step_y(a4),d3
set_ydrift    move.w    d3,drift_y(a4)
              move.l    d6,a2           Restore serial queue
              move.w    d7,d1           Restore new delta
              rts

*
* Routine to translate buttons to key-codes; new buttons in D0
*
button_key    tst.w     d0              Any action?
              beq.s     wits_end
              tst.w     key_flag(a4)    Key emulation on?
              beq.s     wits_end
              move.l    sv_keyq(a6),d1  Is there a key queue?

              beq.s     wits_end        No queue now so ignore
              move.w    d0,d7
              move.l    a2,d6           Save -> Serial queue
              move.l    d1,a2           A2 -> Key queue
              movea.w   $E0.w,a5        A5 = IO.QIN vector
              move.w    button_bits(a4),d1
              btst      #0,d0
              beq.s     not1
              btst      #0,d1           Already down?
              bne.s     not1
              moveq     #button1,d1     Send new keypress
              jsr       (a5)            Ignore 'no room' error
not1          btst      #1,d0           Button two?
              beq.s     not2
              btst      #1,d1           Already down?
              bne.s     not2
              moveq     #button2,d1     Queue it
              jsr       (a5)
*
not2
              ifeq      buttons-3
              btst      #2,d0           Third button?
              beq.s     restore
              btst      #2,d1
              bne.s     restore
              moveq     #button3,d1     Try to queue it
              jsr       (a5)
              endc
*
restore       move.w    d7,d0           Restore new buttons
              movea.l   d6,a2           Restore input queue
wits_end      rts
```

sequent IFNE and ENDC directives ensure that an error occurs if QKEYS is set incorrectly.

The second part of the assembler listing is marked off by a row of stars. It is a re-write of the two button Microsoft mouse driver, extended to allow co-ordinate wrapping and keyboard emulation.

The keyboard emulation and co-ordinate wrapping code was added by re-writing the inner loop of the mouse handler, adding calls to three new subroutines HORIZONTAL, VERTICAL and BUTTON_KEY which issue key-presses as required.

If the version 2 variable WRAP_FLAG is set then an overflow or underflow wraps the co-ordinate to the opposite edge of the screen. I decided it was better to copy and amend the four routines for each axis and mouse-type than to try to put conditional code in each one.

The tweaks may appear complicated but should be reliable; I set out to introduce the smallest number of changes and new lines that would do the job simply. It would be possible to save bytes by merging the vertical and horizontal keyboard routines, but this could introduce new bugs.

The final section should be added before the names and addresses at the end of the assembly file. It contains code to emulate key-presses by putting characters into the current key-buffer as the mouse moves.

The mouse buttons also generate key-codes, corresponding to SPACE, ENTER and ESCAPE (if you use a Mouse Systems compatible pointing device with three buttons). These are the main keys for QRAM and most pointer-based programs.

The code for PTR_KEY is not listed here as it is exactly the same as that for PTR_MAX, PTR_INC and their ilk, except that it adds

24 to A4 - the offset of KEY_FLAG - after calling FIND_POS and before returning via SETWORDS.

Performance

I have tested all these programs with an AXELEN dual-standard switchable mouse, which cost £10 at an All-Formats Computer Fair. The middle button is ignored when it is switched to Microsoft mode. The DIY Toolkit drivers have also been tested with a similar mouse supplied by Dennis Briggs.

Serial mice are widely available at low prices and come with a nine or 15 pin PC connector. If you own a British-made QL you need to replace this with a six-pin plug to suit your SER port. Samsung QLs have a nine-pin connector which may accept the PC plug, but you'll still need to check that the internal links match your chosen QL port. The necessary connections were explained on page 20 of QL World 1993 issue II.9.

Dennis Briggs of Adman Services can build you an adapter for £4 if you do not feel like making up your own. He can also supply PC mice complete with the required QL adapter and the full DIY software Volume for `15 inclusive. Contact Dennis Briggs at Adman Services, 53 Gilpin Road, Admaston, Telford, Shropshire, England, or call him on (0952) 255895.

The DIY mouse handlers check the format of the first byte of each message. If it is inplausible they discard it and wait for a valid one. This means that the handler re-synchronises itself at the start of the next complete message.

If you are using a switchable joystick with Sinclair's serial input chip, you will probably find that the two-button setting is the most reliable. Three-button signals are more likely to lose synchronisation if serial input gets lost. The best way to avoid such problems is to upgrade your system with the Hermes chip, which dramatically improves the performance of the QL serial inputs.

```
100 REMark MODE 8 mini MOUSE_PAINT
110 REMark Use combinations of
120 REMark 3 buttons to pick 8
130 REMark colours for drawing
140 REMark Uses DIY PLOT + DRAW exts
150 :
160 BAUD 1200
170 PTR_ON
180 REMark Full screen
190 OPEN #3,scr_512x256a0x0
200 PTR_MAX 511,255
210 PTR_POS 0,0
220 PTR_KEY 0,0 :REMark Wrap?
230 MODE 8
240 x%=0 : y%=0
250 CLS #3
260 BLINK (x%),(y%)
270 REPeat pain
280   oldx%=x% : oldy%=y%
290   x%=PTR_X% : y%=PTR_Y%
300   BLINK (oldx%),(oldy%)
310   INK #3,BUTTON%(0)
320   PLOT #3,x%,y%
330   BLINK (x%),(y%)
340 END REPeat pain
350 CLOSE #3
360 :
370 DEFine PROCedure BLINK(a,b)
380 REMark Non-destructive '+' cursor
390 OVER #3,-1 : INK #3,7
400 a=a-2 : IF a<0 : a=0
410 IF a>507 : a=507
420 b=b-2 : IF b<0 : b=0
430 IF b>251 : b=251
440 PLOT #3,a+2,b : DRAW #3,a+2,b+4
450 PLOT #3,a,b+2 : DRAW #3,a+4,b+2
460 OVER #3,0
470 END DEFine BLINK
```

The machine-code versions read the serial input queue directly, but can still lose step if Sinclair's old IPC drops bytes on their way from the mouse to Qdos. This is most likely if the QL is busy with a device and cannot read serial bytes for a while. They pile up in the 8049 co-processor, and soon overflow.

Checklist

If your QL is otherwise fine but your serial mouse won't work, check these things first:

(1) Did you load the correct code file for your mouse and adapter? (2) Is the mouse plugged into the correct port for the adapter? (3) Have you set BAUD 1200, PTR_ON, and PTR_KEY 1,1 if needed?

If the mouse is a switchable one:

(4) Is the mouse switched to the right protocol (PC for three buttons or MS for two)?

If the answer to any of these questions is no, reset the QL, check the connections and settings and try again.

If it still won't work, try the mouse on a PC or another QL system. If it works there, but it is erratic on your QL, you need Hermes to fix your QL serial input. This is more likely with old three button mice than with the two-button Microsoft breed.

In either case, **Hermes** is an excellent QL upgrade, ideal for this purpose. It has proper serial inputs, written by a Bulletin board boss who knows RS-232 inside out, so it goes much faster and suits all sorts of serial devices.

A New Mousetrap?

Well, that completes the biggest DIY Toolkit project to date - We've ended up with over seven hundred lines of code, eight different mouse drivers, four SuperBasic demonstrations and six Quill files of documentation. DIY Toolkit Volume I also includes a Turbo-compiled mouse driver with keyboard emulation that works fine with Hermes.

I plan to tackle something a little simpler in the next DIY Toolkit project; as ever I need your suggestions for the column. Readers have sent in dozens of good ideas over the years, and I've implemented most of them. If there's anything new you'd like to see in SuperBasic, please let me know the details and I'll write it for you, and the other DIY Toolkit enthusiasts.

Update

I've made a couple of improvements to the INPUT$ and SET_POS code in QL World 1993 II.8. The DIY Volume has been updated and I want to share the new code with people who type in the assembly listings straight off the page.

The INPUT$ function upsets **Turbo** and **Supercharge** because it does not set A1 on exit. The compilers use the register value, rather than BV.RIP, to find the result. Section 10.7.5 of Adrian Dickens' **QL Advanced User Guide** says that both A1 and BV.RIP should be set appropriately, but QL roms only look at BV.RIP, while the compilers use A1.

I didn't spot this for a while as interpreter tests worked fine and Turbo comes with its own INPUT$ function. To correct the assembly code, add the line MOVEAL D5,A1 after the line annotated SET BV.RIP, near the end of the code for INPUT$.

My article said that SET_POS goes to the end of a file without error if asked to move to a position past the end. This feature was not used in the accompanying SuperBasic, and it seems I forgot it, as users are getting an End Of File report. This code traps any EOF error after the TRAP #3:

```
TRAP_EOF CMP.W    #-10,D0
         BNE.S    RETURN
         MOVEQ    #0,D0
RETURN   RTS
```

Mea Culpa! Thank you for alerting me to this mistake.

SINCLAIR QL WORLD

SUBSCRIBE TO SINCLAIR QL WORLD AT LAST YEARS PRICES AND SAVE £££££'S

£££

UK
£23.40
(12 MONTHS)

EUROPE
£32.90
(12MONTHS)

REST OF WORLD
£40.90
(12 MONTHS)

PLEASE SEND YOUR SUBSCRIPTION FORM TO
SINCLAIR QL WORLD,
THE BLUE BARN, TEW LANE, WOOTTON,
WOODSTOCK, OXON.
OX71HA.
TEL:0993 811181
FAX:0993 811481

I WISH TO SUBSCRIBE TO SINCLAIR QL WORLD WE HAVE HAD TO RAISE THE COVER PRICE OF QL WORLD BUT WILL BE HOLDING LAST YEARS SUBSCRIPTION PRICE FOR THE IMMEDIATE FUTURE ONLY.

Existing subscribers will not be affected.

PLEASE DEBIT MY CREDIT CARD ACCESS/VISA

CARD NUMBER...

EXPIRY DATE

PLEASE SEND ME A 12 MONTH SUBSCRIPTION TO SINCLAIR QL WORLD.

NAME...

ADDRESS...

....

TELEPHONE NUMBER...

SIGNED...

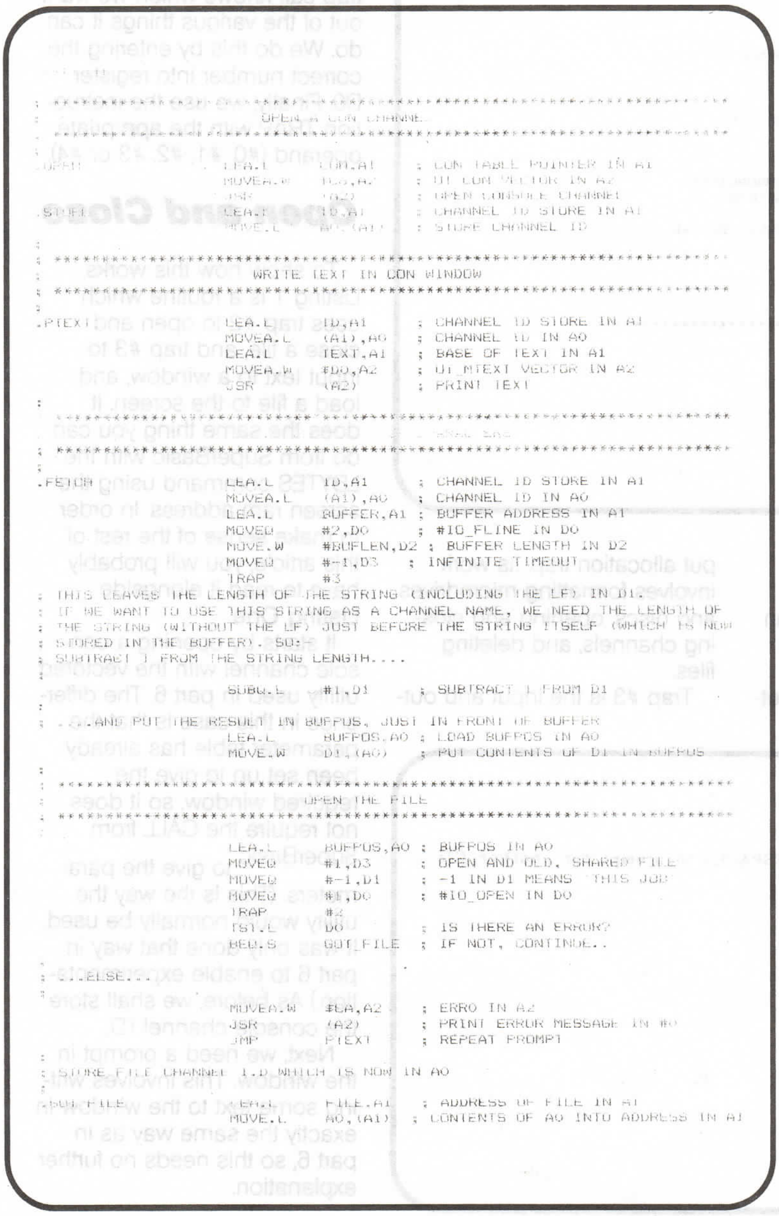
DATE...

STARTING WITH ISSUE...

Machine Code for Beginners

In part 7, Alan Bridewell introduces Trap Calls.

In part 6 we started to look at how we can make our machine code routines do the kind of things we really need them to do: opening channels, writing to the screen, and so on. What we do not do is to actually write the code to do this ourselves because the code already exists in the QL's rom. All we need to do is to access this code and incorporate in our routine.



```
;  **************************************************************
;                     OPEN A CON CHANNEL
;  **************************************************************
;
.OPEN           LEA.L      CON,A1       ; CON TABLE POINTER IN A1
                MOVEA.W    $C6,A2       ; UT CON VECTOR IN A2
                JSR        (A2)         ; OPEN CONSOLE CHANNEL
.STORE          LEA.L      ID,A1        ; CHANNEL ID STORE IN A1
                MOVE.L     A0,(A1)      ; STORE CHANNEL ID
;
;  **************************************************************
;                   WRITE TEXT IN CON WINDOW
;  **************************************************************
;
.PTEXT          LEA.L      ID,A1        ; CHANNEL ID STORE IN A1
                MOVEA.L    (A1),A0      ; CHANNEL ID IN A0
                LEA.L      TEXT,A1      ; BASE OF TEXT IN A1
                MOVEA.W    $D0,A2       ; UT_MTEXT VECTOR IN A2
                JSR        (A2)         ; PRINT TEXT
;
;  **************************************************************
;                          READ LINE
;  **************************************************************
;
.FETCH          LEA.L      ID,A1        ; CHANNEL ID STORE IN A1
                MOVEA.L    (A1),A0      ; CHANNEL ID IN A0
                LEA.L      BUFFER,A1 ; BUFFER ADDRESS IN A1
                MOVEQ      #2,D0        ; #IO_FLINE IN D0
                MOVE.W     #BUFLEN,D2 ; BUFFER LENGTH IN D2
                MOVEQ      #-1,D3       ; INFINITE TIMEOUT
                TRAP       #3
; THIS LEAVES THE LENGTH OF THE STRING (INCLUDING THE LF) IN D1.
; IF WE WANT TO USE THIS STRING AS A CHANNEL NAME, WE NEED THE LENGTH OF
; THE STRING (WITHOUT THE LF) JUST BEFORE THE STRING ITSELF (WHICH IS NOW
; STORED IN THE BUFFER). SO:-
; SUBTRACT 1 FROM THE STRING LENGTH....
;
                SUBQ.L     #1,D1        ; SUBTRACT 1 FROM D1
;
; ...AND PUT THE RESULT IN BUFPOS, JUST IN FRONT OF BUFFER
                LEA.L      BUFPOS,A0 ; LOAD BUFPOS IN A0
                MOVE.W     D1,(A0)      ; PUT CONTENTS OF D1 IN BUFPOS
;
;  **************************************************************
;                         OPEN THE FILE
;  **************************************************************
;
                LEA.L      BUFPOS,A0 ; BUFPOS IN A0
                MOVEQ      #1,D3        ; OPEN AND OLD, SHARED FILE
                MOVEQ      #-1,D1       ; -1 IN D1 MEANS  THIS JOB
                MOVEQ      #1,D0        ; #IO_OPEN IN D0
                TRAP       #2
                TST.L      D0           ; IS THERE AN ERROR?
                BEQ.S      GOT_FILE  ; IF NOT, CONTINUE..
;
; ...ELSE...
;
                MOVEA.W    $CA,A2       ; ERRO IN A2
                JSR        (A2)         ; PRINT ERROR MESSAGE IN D0
                JMP        PTEXT        ; REPEAT PROMPT
;
; STORE FILE CHANNEL I.D WHICH IS NOW IN A0
;
.GOT_FILE       LEA.L      FILE,A1      ; ADDRESS OF FILE IN A1
                MOVE.L     A0,(A1)    ; CONTENTS OF A0 INTO ADDRESS IN A1
```

Rom Routines

The routines in the rom are of two types, called "vectored utilities" and "trap calls". In part 6 we looked at vectored utilities, so now we shall move on to trap calls. As far as running your programs is concerned there is no noticable difference between the two types of routine. If you have the facility to trace through your machine code, one instruction at a time, you will notice one difference. When you use a vectored utility, you will trace through the whole subroutine as though it was part of your code. When you use a trap call, it will not be traced.

As far as you, as programmer, are concerned, the big difference between vectored utilities and trap calls is the way you access them in your program. However, we should look at what exactly a trap call is, although much of the detail of this is very hard going, and not really needed unless you intend to do some high powered programming. I shall only cover it in outline.

The microprocessor in the QL can operate in two modes, called "user mode" and "supervisor mode". User mode is what programs normally use, and everything mentioned is this series of articles is in user mode. When the processor is in supervisor mode, the program in user mode is temporarily suspended, while special supervisor mode routines are run. These routines can be exactly the same as user mode routines, but can also contain some special "privileged" instructions.

Supervisor Mode

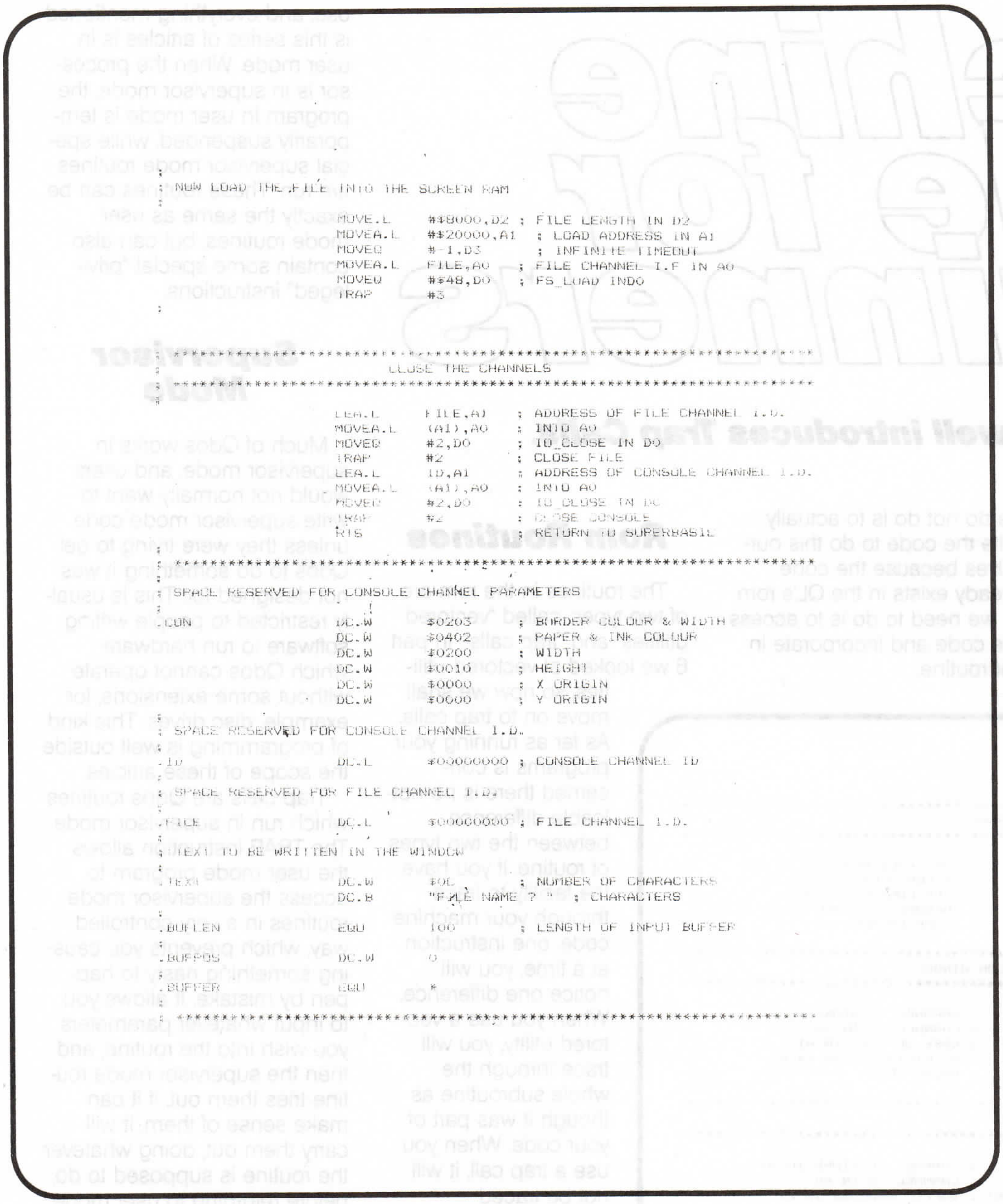
Much of Qdos works in supervisor mode, and users would not normally want to write supervisor mode code unless they were trying to get Qdos to do something it was not designed for. This is usually restricted to people writing software to run hardware which Qdos cannot operate without some extensions, for example, disc drives. This kind of programming is well outside the scope of these articles.

Trap calls are Qdos routines which run in supervisor mode. The TRAP instruction allows the user mode program to access the supervisor mode routines in a very controlled way, which prevents you causing something nasty to happen by mistake. It allows you to input whatever parameters you wish into the routine, and then the supervisor mode routine tries them out. If it can make sense of them, it will carry them out, doing whatever the routine is supposed to do, before returning to user mode at the next instruction in your code, with no error code.

If it cannot make sense of them, it will also return to user mode at the next instruction in your code, but this time returning with an error code in register D0. What this means is that your mistake will not crash the machine, as long as you take appropriate action on receiving the error code.

16 Calls

The microprocessor recognises 16 different trap calls, numbered #0 to #15, but the QL only uses the first five, #0 to #4. Trap #0 is concerned with going into supervisor mode, and unless you are an expert or are intent on crashing your machine and maybe



```
;
; NOW LOAD THE FILE INTO THE SCREEN RAM
;
              MOVE.L     #$8000,D2 ; FILE LENGTH IN D2
              MOVEA.L    #$20000,A1  ; LOAD ADDRESS IN A1
              MOVEQ      #-1,D3      ; INFINITE TIMEOUT
              MOVEA.L    FILE,A0     ; FILE CHANNEL I.F IN A0
              MOVEQ      #$48,D0     ; FS_LOAD INDO
              TRAP       #3
;

;*******************************************************************
;                    CLOSE THE CHANNELS
;*******************************************************************
;
              LEA.L      FILE,A1     ; ADDRESS OF FILE CHANNEL I.D.
              MOVEA.L    (A1),A0     ; INTO A0
              MOVEQ      #2,D0       ; IO_CLOSE IN D0
              TRAP       #2          ; CLOSE FILE
              LEA.L      ID,A1       ; ADDRESS OF CONSOLE CHANNEL I.D.
              MOVEA.L    (A1),A0     ; INTO A0
              MOVEQ      #2,D0       ; IO_CLOSE IN DC
              TRAP       #2          ; CLOSE CONSOLE
              RTS                    ; RETURN TO SUPERBASIC
;
;*******************************************************************
;
; SPACE RESERVED FOR CONSOLE CHANNEL PARAMETERS
;
.CON                 DC.W       $0203      ; BORDER COLOUR & WIDTH
                     DC.W       $0402      ; PAPER & INK COLOUR
                     DC.W       $0200      ; WIDTH
                     DC.W       $0010      ; HEIGHT
                     DC.W       $0000      ; X ORIGIN
                     DC.W       $0000      ; Y ORIGIN
;
; SPACE RESERVED FOR CONSOLE CHANNEL I.D.
;
.ID                  DC.L       $00000000 ; CONSOLE CHANNEL ID
;
; SPACE RESERVED FOR FILE CHANNEL I.D.
;
.FILE                DC.L       $00000000 ; FILE CHANNEL I.D.
;
; TEXT TO BE WRITTEN IN THE WINDOW
;
.TEXT                DC.W       $0C        ; NUMBER OF CHARACTERS
                     DC.B       "FILE NAME ? " ; CHARACTERS
;
.BUFLEN              EQU        100        ; LENGTH OF INPUT BUFFER
;
.BUFPOS              DC.W       0
;
.BUFFER              EQU        *
;
; *******************************************************************
```

even damaging your hardware, you should leave well alone. I shall not mention it again.

Trap #1 is the manager trap. It is concerned with all the memory allocation, priority of mutitasking jobs, keyboard access, clock, and various other bits and pieces that can be grouped under the name "management".

Trap #2 is the input and output allocation trap. Its work involves formatting microdrives and discs, opening and closing channels, and deleting files.

Trap #3 is the input and output utilisation trap. Having a channel open already, this trap can do whatever can be done to the channel. This will, of course, depend on what type of channel it is, a screen or console, a new file to be written to, an old file to be read, or maybe overwritten. Most of the interesting things you might want to do are done by trap #3 calls.

Trap #4 is concerned with making adjustments for certain addressing modes when using SuperBasic. It need not concern us here.

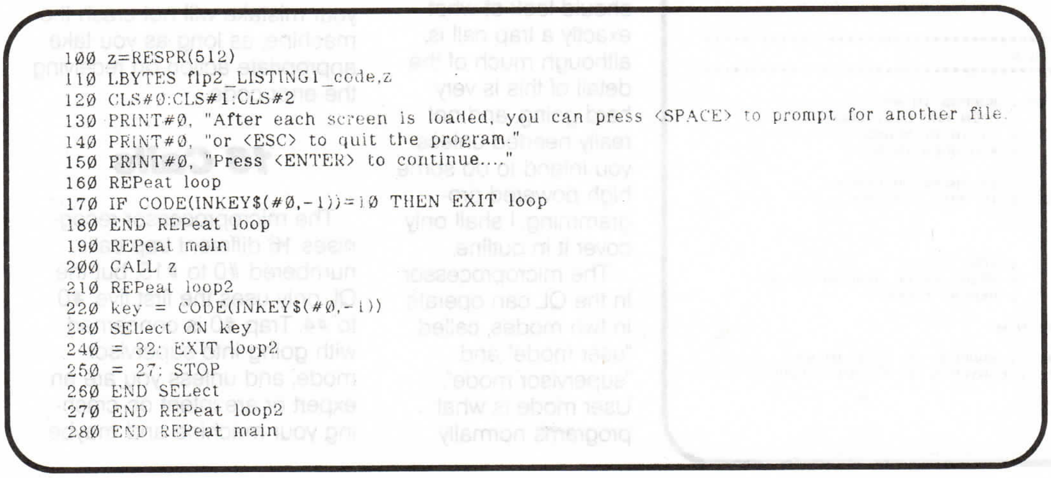
So how do we use these trap calls? The method is very similar to using the vectored utilities mentioned in part 6. First we must make sure all the necessary parameters are in the correct registers, and any data or parameter tables have been correctly set up. We then have to make sure the trap call knows which we want out of the various things it can do. We do this by entering the correct number into register D0. Finally, we use the instruction TRAP with the apprpriate operand (#0, #1, #2, #3 or #4).

Open and Close

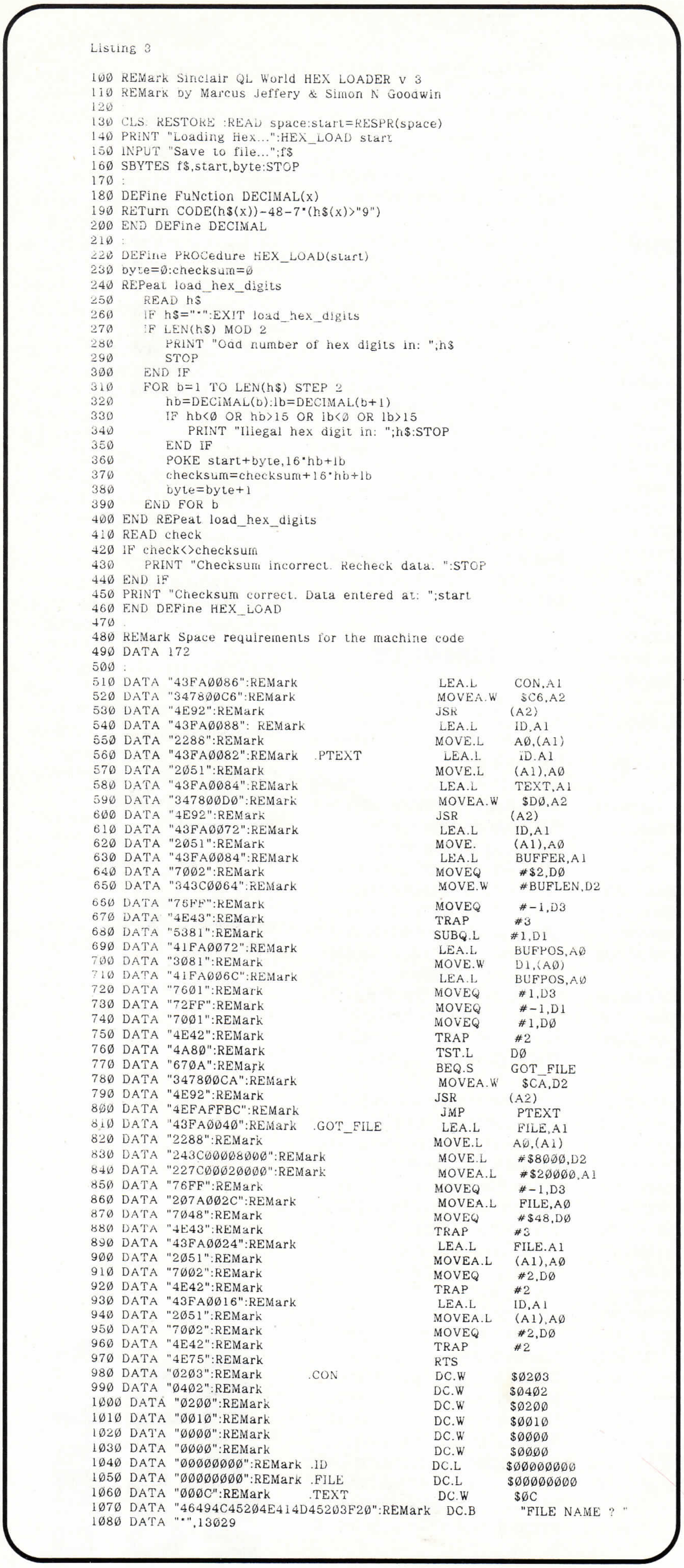
To show how this works, Listing 1 is a routine which uses trap #2 to open and close a file, and trap #3 to input text to a window, and load a file to the screen. It does the same thing you can do from SuperBasic with the LBYTES command using the screen ram address. In order to make sense of the rest of this article, you will probably have to read it alongside **Listing One.**

It starts by opening a console channel with the vectored utility used in part 6. The difference in this case is that the parameter table has already been set up to give the required window, so it does not require the CALL from SuperBasic to give the parameters. (This is the way the utility would normally be used. It was only done that way in part 6 to enable experimentation.) As before, we shall store the console channel I.D.

Next, we need a prompt in the window. This involves writing some text to the window in exactly the same way as in part 6, so this needs no further explanation.



```
100 z=RESPR(512)
110 LBYTES flp2_LISTING1_code,z
120 CLS#0:CLS#1:CLS#2
130 PRINT#0, "After each screen is loaded, you can press <SPACE> to prompt for another file.
140 PRINT#0, "or <ESC> to quit the program."
150 PRINT#0, "Press <ENTER> to continue..."
160 REPeat loop
170 IF CODE(INKEY$(#0,-1))=10 THEN EXIT loop
180 END REPeat loop
190 REPeat main
200 CALL z
210 REPeat loop2
220 key = CODE(INKEY$(#0,-1))
230 SELect ON key
240 = 32: EXIT loop2
250 = 27: STOP
260 END SELect
270 END REPeat loop2
280 END REPeat main
```



Listing 3

```
100 REMark Sinclair QL World HEX LOADER v 3
110 REMark by Marcus Jeffery & Simon N Goodwin
120
130 CLS: RESTORE :READ space:start=RESPR(space)
140 PRINT "Loading Hex...":HEX_LOAD start
150 INPUT "Save to file...";f$
160 SBYTES f$,start,byte:STOP
170 :
180 DEFine FuNction DECIMAL(x)
190 RETurn CODE(h$(x))-48-7*(h$(x)>"9")
200 END DEFine DECIMAL
210 :
220 DEFine PROCedure HEX_LOAD(start)
230 byte=0:checksum=0
240 REPeat load_hex_digits
250    READ h$
260    IF h$="*":EXIT load_hex_digits
270    IF LEN(h$) MOD 2
280       PRINT "Odd number of hex digits in: ";h$
290       STOP
300    END IF
310    FOR b=1 TO LEN(h$) STEP 2
320       hb=DECIMAL(b):lb=DECIMAL(b+1)
330       IF hb<0 OR hb>15 OR lb<0 OR lb>15
340          PRINT "Illegal hex digit in: ";h$:STOP
350       END IF
360       POKE start+byte,16*hb+lb
370       checksum=checksum+16*hb+lb
380       byte=byte+1
390    END FOR b
400 END REPeat load_hex_digits
410 READ check
420 IF check<>checksum
430    PRINT "Checksum incorrect. Recheck data. ":STOP
440 END IF
450 PRINT "Checksum correct. Data entered at: ";start
460 END DEFine HEX_LOAD
470 .
480 REMark Space requirements for the machine code
490 DATA 172
500 :
510 DATA "43FA0086":REMark                LEA.L     CON,A1
520 DATA "347800C6":REMark                MOVEA.W   $C6,A2
530 DATA "4E92":REMark                    JSR       (A2)
540 DATA "43FA0088": REMark               LEA.L     ID,A1
550 DATA "2288":REMark                    MOVE.L    A0,(A1)
560 DATA "43FA0082":REMark  .PTEXT        LEA.L     ID.A1
570 DATA "2051":REMark                    MOVE.L    (A1),A0
580 DATA "43FA0084":REMark                LEA.L     TEXT,A1
590 DATA "347800D0":REMark                MOVEA.W   $D0,A2
600 DATA "4E92":REMark                    JSR       (A2)
610 DATA "43FA0072":REMark                LEA.L     ID,A1
620 DATA "2051":REMark                    MOVE.     (A1),A0
630 DATA "43FA0084":REMark                LEA.L     BUFFER,A1
640 DATA "7002":REMark                    MOVEQ     #$2,D0
650 DATA "343C0064":REMark                MOVE.W    #BUFLEN,D2
660 DATA "76FF":REMark                    MOVEQ     #-1,D3
670 DATA "4E43":REMark                    TRAP      #3
680 DATA "5381":REMark                    SUBQ.L    #1,D1
690 DATA "41FA0072":REMark                LEA.L     BUFPOS,A0
700 DATA "3081":REMark                    MOVE.W    D1,(A0)
710 DATA "41FA006C":REMark                LEA.L     BUFPOS,A0
720 DATA "7601":REMark                    MOVEQ     #1,D3
730 DATA "72FF":REMark                    MOVEQ     #-1,D1
740 DATA "7001":REMark                    MOVEQ     #1,D0
750 DATA "4E42":REMark                    TRAP      #2
760 DATA "4A80":REMark                    TST.L     D0
770 DATA "670A":REMark                    BEQ.S     GOT_FILE
780 DATA "347800CA":REMark                MOVEA.W   $CA,D2
790 DATA "4E92":REMark                    JSR       (A2)
800 DATA "4EFAFFBC":REMark                JMP       PTEXT
810 DATA "43FA0040":REMark  .GOT_FILE     LEA.L     FILE,A1
820 DATA "2288":REMark                    MOVE.L    A0,(A1)
830 DATA "243C00008000":REMark            MOVE.L    #$8000,D2
840 DATA "227C00020000":REMark            MOVEA.L   #$20000,A1
850 DATA "76FF":REMark                    MOVEQ     #-1,D3
860 DATA "207A002C":REMark                MOVEA.L   FILE,A0
870 DATA "7048":REMark                    MOVEQ     #$48,D0
880 DATA "4E43":REMark                    TRAP      #3
890 DATA "43FA0024":REMark                LEA.L     FILE.A1
900 DATA "2051":REMark                    MOVEA.L   (A1),A0
910 DATA "7002":REMark                    MOVEQ     #2,D0
920 DATA "4E42":REMark                    TRAP      #2
930 DATA "43FA0016":REMark                LEA.L     ID,A1
940 DATA "2051":REMark                    MOVEA.L   (A1),A0
950 DATA "7002":REMark                    MOVEQ     #2,D0
960 DATA "4E42":REMark                    TRAP      #2
970 DATA "4E75":REMark                    RTS
980 DATA "0203":REMark      .CON          DC.W      $0203
990 DATA "0402":REMark                    DC.W      $0402
1000 DATA "0200":REMark                   DC.W      $0200
1010 DATA "0010":REMark                   DC.W      $0010
1020 DATA "0000":REMark                   DC.W      $0000
1030 DATA "0000":REMark                   DC.W      $0000
1040 DATA "00000000":REMark .ID           DC.L      $00000000
1050 DATA "00000000":REMark .FILE         DC.L      $00000000
1060 DATA "000C":REMark     .TEXT         DC.W      $0C
1070 DATA "46494C45204E414D45203F20":REMark DC.B    "FILE NAME ? "
1080 DATA "*",13029
```

The program now needs to input the name of the file by fetching a line of characters from the keyboard. This is part of input/output utilisation and involves a trap #3 call. Before the trap call can do this it needs the channel ID of the console channel in register A0. It also needs the address of a buffer to store the text in register A1 with the length of the buffer in register D2. In order to make sure the characters going into the buffer do not overwrite your program or parameters, the buffer should come at the end.

In order that the trap call knows which bit of input/output utilisation it is to do, it needs a number in register D0 to tell it. The number for fetching a line of text is #2. (This number is usually given the label IO_FLINE, aad your assembler may accept this label instaed of the number.)

Finally, trap #3 requires a "timeout" in register D3. This is the time, in fiftieths of a second, the call will wait until the utilisation is completed. It will also accept zero to mean "don't wait at all" and -1 to mean "wait for ever if necessary". Frankly, I have always found a timeout of -1 works fine, and that is what we shall use.

With all the parameters correctly positioned in the registers, we make the trap call with the line

TRAP #3

At this stage, we could check register D0 for an error code. Apart from an error in typing in the listing, the only possible error is a file name which is too long for the buffer, which would give a "buffer overflow" error. Since we have made the buffer 100 bytes long, this is very unlikely to happen. I shall come back to this point later.

Juggling

With a successful trap call, the buffer should now contain the name of the screen file we wish to open. We now have to use that name to actually open the file. However, in order to do this, we need to do a little juggling, because the buffer does not contain the file name in quite the correct form.

The trap #3 call will leave the file name in the buffer. But the string of characters will contain the line feed character we get when we press Enter at the end of the line. It also leaves the length of the string (including the line feed) in register D1.

Now the trap #2 call we will use to open the file requires the file name in a slightly different form. It expects the file name to begin with a word (two bytes) which give the number of characters in the file name, followed by the file name itself, but without the line feed character at the end. To get over this difference, we reserve a word (two bytes) immediately before the buffer. We have labelled this word BUFPOS. We subtract one from the character count in register D1, and then move the resulting number into BUFPOS. If we now use BUFPOS rather than BUFFER as the name of the file to be opened, it will be in the correct form, starting with the word giving the number of characters, followed by the characters themselves. Because we have subtracted one from the character count, it will ignore the line feed character at the end.

The next step is to open the file with the trap #2 call. Before we can do this we need to put all the appropriate data into the registers. Register A0 must contain the address of the file name, BUFPOS. Register D3 must contain a number depending on what kind of file is to be opened. The file is an old one, and we may be sharing it (it may also be read by another multitasking program at the same time), and this requires a one in register D3.

Job ID

Register D1 requires the job ID of the job using the channel. It is possible for one multitasking job to open a file for another, in which case, the appropriate job ID would need to be found. However, the trap call will accept -1 in D1 to mean "this job".

Finally, it needs a one in register D1 to tell it open the file. The number one has the label IO_OPEN, which your assembler may accept in place of one. We can now make the trap call with the instruction

TRAP #2

This time we must check for an error after returning from the trap call. Even if our code is correct (and it should be) there is the very good chance that the file name typed in is incorrect so that no channel has been opened, in which case we need to take appropriate action.

There is a general rule here. If the parameters are written into the program, there should be no error, and so no need to check. But if any of the parameters come from user input, an error is possible and should be checked for.

In part 6 we tested for an error return by comparing register D0 with zero using the instruction

```
CMPI.W  #$00,D0
```

CMPI is a general instruction for finding out if we have any particular value, or a bigger or smaller value. But when we do error testing, we are only interested in whether we have a zero or a negative number. For this we have another instruction, TST, which is rather more concise, and so runs faster. Normally, we should use TST for error testing. I only used CMPI before because it is a very useful instruction in many situations, and is well worth being familiar with.

What TST does is simply tell us if the number we have is zero or negative by setting the appropriate flag in the status register. This is exactly what we need here.

Branch and Hang?

After the TST instruction, a BEQ instruction will branch if the zero flag is set (ie no error) to the next bit of code when everything goes to plan. In our program this has the label GOTFILE. But what do we do if an error is detected, and the program goes to the next instruction, instead of branching to GOTFILE? We could simply abort the program and return to SuperBasic with a RTS instruction. This would not crash the QL, but it would drop you out of the program, which might be just as annoying. There is a much better alternative.

There are really only two possibilities for the error. It could have been an error in typing in the file name, or it could be that the file you want is not actually on the medium in that particular drive. Either way, the best thing to do is to print an appropriate error message on the screen, then loop back to allow you to retype the file name. This is what the next three lines do.

When the file is successfully opened, the file channel I.D. is left in register A0, so we store it in a space reserved for it with the label FILE.

The next job is to load the file into the screen ram with a trap #3 call. To do this we need the file length in register D2. If it is a normal screen file it will be 32K long, which is $8000 in hex. The address to start loading must be put in register A1, and this is the start address of the screen RAM, $20000. To indicate an infinite timeout we put -1 in register D3. To show that what we are doing is loading a file, we put $48 in register D0. ($48 has the label FS_LOAD which your assembler might accept in place of $48.) This is finally followed by the trap call with the instruction

TRAP #3

If all has gone well, your saved screen file should load into the screen ram and appear on your screen.

At this point we should tidy things up by closing the channels we have now finished with, that is, the file channel and the console window channel. This is done in each case by loading the channel ID into register A0, loading #2 into register D0 (#2 has the label IO_CLOSE which your assembler may accept) and then making the trap #2 call. After all this we return to SuperBasic with RTS.

After the code come all the various spaces reserved for parameters and data. These can come in any order, except that the BUFFER should come last (and, of course, BUFPOS must come immediately before it.)

Listings

Listing Two is a small SuperBasic routine to run the code. It needs no explanation. Listing 3 is the code contained in Marcus' and Simon's Hex Loader for those who do not have an assembler.

Having got the thing to work, the next job is to experiment with it to show that you really do understand what is going on. You could do simple things like changing the window parameters, or changing the prompt, but that has no impact on the trap calls. There is a much more obvious improvement you could try.

As the program stands, you must eventually give it a usable screen file name, otherwise it will repeat the prompt forever, or crash the system when you try to load the wrong file type. Either way, if you can't give it a usable file name, you will have to reset the QL to get back to SuperBasic.

One way out of this is to test for a zero string length before trying to open the file, and jumping to a RTS instruction if a zero string is found. This will enable you to leave the program by pressing Enter at the prompt without having to give a file name. This will make it a lot more user friendly.

Subtle ...

For something more subtle you can look into this. In this program we have religiously saved each channel ID and loaded the channel ID back into the registers when they are needed. While in general this process is necessary, if we are using a channel immediately it is opened, the channel ID is already in the register and does not need reloading. Also, if we are not going to need it again later, there is no need to store it. It is an interesting exercise to see how many redundant lines of storing and reloading channel IDs can be removed and still allow the program to run. If you can do this successfully, then you probably have a good grasp of how the program works.

This article only gives a glimpse of what can be done with trap calls. To go further would need long lists of all the things the trap calls do and what needs to be in each register for each operation. I have already mentioned published works which contain this information. I would simply add to this list my own series of articles "Systematic M/C Programming", which were printed in QL World from October 1991 to September 1992. They contain a lot of examples of the most useful trap calls, particularly the article in February 1992, which covered most of the trap #3 calls.

Happy coding!

Machine Code part 7

MICRO ADS

SINCLAIR QL + twin floppy drives, 256k memory, separate keyboard, many games £125 0491 681821

QL SYSTEM comprising 2No QL's, colour monitors, 2No 3 1/2" disc drives, trumpcard, silver-reed plotter/printer, thermal printer; software, taskmaster, spellbound, lightning, Techniql etc. Includes all leads £220

Plus. Psion llcm organiser including letter organiser, QL comms-link, Data paks £60

Prices exclude postage. Tel eve 051 726 9801

QL unexpanded AH version PSU Psion software manual etc. Any offers Miracle Systems QL modem £20. Phone: John 0689 855444

JS QL, Metal case, PC keyboard and I/face, twin 3-1/2" d/drives and OPD m/drives, some software, new colour monitor £250 ono. Also Tandata modem 3, CQVI Digitiser and more. Tel: (Derby) 0332 677746

QL unexpanded JM version, shinwa CPA 80Q printer, joystick, Software including Psion suite, Eye-Q and Games, 40 MDV's, Manuals etc, £150 Tel:0256 764006 evenings only

TWO QL's, microvitec monitors, two twin 3 1/2 drives, goldcard, trumpcard conq.goldse, multidiscover, textidy, Sinclair user/world from 8/85 £250.00 Buyer to collect. Keith 081 302 2266, 081 402 1858 (eve) Beckenham Kent)

WITH REGRET, Offers for complete QL System. Two Computers 14" colour monitor, double disk drive, ice, original software plus more. Books, Manuals. Psion organiser and printer. Phone for details 0257 265700

Professional poolster REVOLUTIONARY NEW SYSTEM

COMBINES

Sound logic with A.I. to win £ hundreds £
Analysis of Bookies Factors for matches
Common sense improves chances of Mega-Win
UNIQUE ENTRY SYSTEM
Takes only a few minutes each week
Over 300 teams in data-bank.
Minor league teams automatically added.
Prior to PC release @ £99.00
To QL users offered at SPECIAL PRICE £49

THIS MONTH

MONEY BACK GUARANTEE - Make a profit in your first full season or your money back.

256K ROM minimum. Discs Only. State 3.5 or 5.25

CHEQUE OR P.O. TO S.CANTON, OCEAN COMPUTER SERVICES, PO BOX 68, NORTH DO. Manchester M8 6NJ
Tel. 061 740 9002

CLASSIFIED

Please send coupon to: Advertising Department Sinclair/QL World, The Blue Barn, Wootton, Woodstock, Oxon. OX7 1HA. Please place this Classified Advertisement in the next issue of Sincair/QL world.

NAME ....

ADDRESS ....

....

DAYTIME TELEPHONE NO ....

I enclose a cheque/P.O. to the value of £17.65 (inc. VAT) for a lineage advertisement of no more than 30 words. Please make cheques payable to: Arcwind Ltd.

DJC Dilwyn Jones Computing

41 BRO EMRYS, TAL-Y-BONT, BANGOR,
GWYNEDD, LL57 3YT, GREAT BRITAIN
FAX/TEL: (0248) 354023

DATABASE SOFTWARE

COCKTAILS WAITER	£10 00
DATA DESIGN 3	£60.00
DATA DESIGN API	£20.00
DBEASY	£15.00
DBPROGS	£15.00
FLASHBACK	£25.00
FLASHBACK SE	£40.00
SUPER DISK INDEXER	£12.00

DTP & CLIPART

PAGE DESIGNER 3	£40.00
Available at last!	
UPGRADE FROM PD2	£25.00
SCANNED CLIPART 1 (U)	£10.00
SCANNED CLIPART 2 (U)	£10.00
THE CLIPART (U)	£12.00

FILE HANDLING

FILEMASTER	£12.00
FILES 2 (U)	£12.00
4MATTER & LOCKSMITHE	£23.50
LOCKSMITHE (U)	£14.95
MDV TOOLCHEST	£14.95
THE GOPHER (U)	£12.00
WINBACK 2	£25.00

FILE TRANSFER

CONVERT-PCX	£10.00
DISCOVER	£20.00
MULTI-DISCOVER	£30.00
OPD INTERCHANGE	£15.00
QL-PC FILESERVER	£24.50
SToQL	£24.50
TEXTIDY	£15.00

GAMES & LEISURE

FIVE GAME PACK (U)	£12.50
FLEET TACTICAL COMMAND (U)	£39.95
FTC DATA PRINTER	£9.95
FLIGHTDECK (U)	£15.00
FUGITIVE	£9.95
GREYWOLF	£12.50
OPEN GOLF	£12.50
QUESTION MASTER (U)	£10.00
QUIZ MASTER 2 (U)	£12.50
RETURN TO EDEN	£17.50
SOLITUDE (U)	£15.00
SQUIDGY ROUND THE WORLD (U)	£12.50

LABELLING

ADDRESS BOOK&LABEL PRINTER	£15.00
SUPER DISK LABELLER	£10.00

GENEALOGY

GENEALOGIST 3	£60.00
new pointer driven version	
Upgrade 2nd Edition	£33.00
upgrade other versions, ask	
GENEALOGIST 2ND EDITION	£30.00
BUDGET 128K GENEALOGIST (U	£12.00

GRAPHICS

IMAGE-D (U)	£10.00
IMAGE PROCESSOR	£15.00
LINE DESIGN	£100.00
PAINTER	£25.00
PICTUREMASTER	£15.00
PICTUREMASTER PLUS	£20.00
QRACTAL	£20.00
QUICK MANDELBROT 3 (U)	£15.00
SCREEN COMPRESSION	£10.00
SCREEN SNATCHER (U)	£10.00
SIMPLE VIDEO TITLES	£5.00
TRANS24 (U)	£10.00

HARDWARE

MICRO PROCESS CONTROLLER	
new redesigned version	£64.50
MPC SOFTWARE	£9.95
NETWORK PROVER	£4.00
QPOWER REGULATOR	£24.95
(unsuitable Gold Card)	
SERMOUSE	£40.00
VOICE ANALYSER-New!	
Ask for details and price	

Add £2 50 postage for MPC,
SERMOUSE AND VOICE ANALYSER

MAGAZINES

QREVIEW	£2.00
QL TECHNICAL REVIEW	£2.00
issues 1-8 available	
QL ADVENTURER'S FORUM	£1.75
Issues 1-9 available	
QL LEISURE REVIEW	£2.00
issues 1-2 available	

(UK prices only shown)

PROGRAMMING

BASIC REPORTER (U)	£10.00
DISA	£29.00
DJTOOLKIT (U)	£10.00
EASYPTR 3 PART 1	£40.50
EASYPTR 3 PART 2	£20.00
EASYPTR 3 PART 3	£20.00
MEGATOOLKIT DISK (U)	£25.00
MEGATOOLKIT EPROM (U)	£40.00
QLIBERATOR 3 36	£50.00
BUDGET QLIBERATOR (U)	£25.00
QLOAD AND QREF (U)	£15.00
S EDIT EDITOR	£20.00

SCREEN DUMPS

SIDEWINDER PLUS	£24.95

SPREADSHEET PRINTING

SIDEWRITER (U)	£15.00
3D TERRAIN	£12.50

SUNDRIES

3 5" DSDD DISKS each	£0.40
3 5" DSHD DISKS each	£0.70
3 5" DISK LABELS (roll 100)	£2.00
3 5" LABELS (printer roll)	£2.50
ADDRESS LABELS (roll 100)	£2.00
MICRODRIVE CARTRIDGES	£2.50
MDV LABELS (roll 100)	£2.00
MOUSE MAT	£2.50
3 5" DISK DIVIDERS (20)	£3.00

Add £0 50 postage for labels or mouse mat if only ordering those, or £2 50 postage for disks or disk box divider sets

TEXT

BANTER BANNERS	£25.00
BIBLE TEXT	£20.00
QTYP2	£29.95
QUICK POSTERS (U)	£10.00
ROB ROY PACK (U)	£10.00
SPELLBOUND	£30.00
SPELLBOUND S E	£50.00
TEXTIDY	£15.00
TEXT 'N' GRAPHIX	£20.00

OTHER SOFTWARE

HOME BUDGET (U)	£20.00
PRINTERMASTER (U)	£20.00
QPAC1	£19.95
QPAC2	£39.95
THE Pointer Environment package!	
QTOP	£29.95
SCREEN DAZZLER	£15.00
SCREEN ECONOMISER (U)	£10.00
SLOWGOLD (U)	£5.00
SPEEDSCREEN (U)	£15.00
SPEEDSCREEN EPROM (U)	£30.00
TASKMASTER	£25.00
VISION MIXER 1	£10.00
VISION MIXER PLUS	£22.50

CALL OR WRITE FOR A FREE COPY OF OUR QL SOFTWARE CATALOGUE DETAILING AROUND 100 QL PRODUCTS

PLEASE NOTE: (U) ABOVE MEANS THAT THE SOFTWARE IS SUITABLE FOR USE ON A 128K QL

TERMS: POSTAGE-software sent post free to UK, overseas add £1 00 per program (maximum £3 00) Floppy disks, SERmouse, etc add £2 50 postage (see above) PAYMENT - IN UK currency (pounds sterling) only, please Valid methods of payment are cheque drawn on UK branch of bank or building society, Eurocheque, Postal Order, cash (send by registered post) or by credit card - Visa, Access, Mastercard and Eurocard accepted Please make cheques etc payable to DILWYN JONES COMPUTING Minimum order value £5 00 (due to bank charges) Goods remain property of Dilwyn Jones Computing until paid for in full Orders normally sent out within a few days of receipt, we will attempt to advise if any delay anticipated due to stock problems Orders can be accepted by telephone if paid for with credit card Orders paid with credit card can only be sent to cardholder's address under card company rules FAX - We now have a Fax machine on our usual number (0248) 354023